IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of	:	
HAWKES, RYCHARDE JEFFERY *et al.*	:	Confirmation No. 1484
U.S. Patent Application No. 09/977,501	:	Group Art Unit: 2143
Filed: October 16, 2001	:	Examiner: Jude Jean Gilles

For: CONTENT PROVIDER ENTITY FOR COMMUNICATION SESSION

Declaration Under 37 C.F.R. §1.131

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I, Colin Andrew Low, based on information and belief, do hereby declare that:

1. I am an inventor of the invention described in U.S. Patent application serial number 09/977,501 (the '501 application) and filed on October 16, 2001 and described in U.K. Patent Application No. 0025454.0 (the U.K. Application) filed on October 17, 2000 and from which the '501 application claims priority.

2. At the time of the invention, I was employed by Hewlett-Packard Company (HP). I am no longer employed by HP as of February 3, 2006.

3. I conceived at least a portion of the invention claimed in claims 19 and 36 of the above-identified patent application prior to February 29, 2000 in the United Kingdom and diligently worked the invention until a constructive reduction to practice in the United Kingdom as demonstrated by the Attached Exhibits and the above-noted '501 application.

4. Prior to February 29, 2000, I contributed to the generation of a disclosure of the invention in the form of a document entitled, "A Web Interaction System." Attached as Exhibit A is a copy of the disclosure document. Exhibit A describes a Web Interaction system created for CRM, E-commerce, and customer interaction applications and services.

5. Prior to February 29, 2000, I contributed to the generation of an additional

disclosure of the invention in the form of a document outlining potential patent claims entitled, "Appendix 1 - Potential Patent Claims." Attached as Exhibit B is a copy of the document.

6. Prior to February 29, 2000, I, generated and transmitted an email including exhibits A and B to Mr. Robert Squibbs of HP for use in preparing the '501 application. Attached as Exhibit C is a copy of the email message.

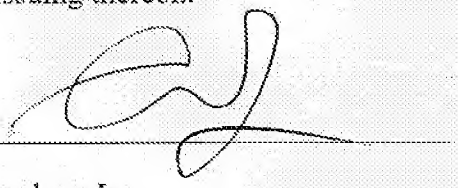
7. During the period between just prior to February 29, 2000 until October 17, 2000, I worked with Mr. Squibbs and provided information regarding the invention assisting Mr. Squibbs to draft the U.K. Application. As evidence of my interaction over this time period, attached Exhibit D is a copy of another detailed document describing information related to the Exhibit B document to which I contributed. Attached Exhibit E is a screenshot of the Microsoft Word file properties dialog displaying the date of creation of the file corresponding to Exhibit D. As further evidence of my interaction over this time period, attached Exhibit F is a copy of a version of the specification and drawings which I received from Mr. Squibbs during the afore-mentioned time period. During the afore-mentioned time period, I reviewed the application in order to provide comments and correction to Mr. Squibbs. During the afore-mentioned time period, I received an email message from Mr. Squibbs requesting me to review a revised version of the claims of the application prior to filing of the application. Attached as Exhibit G is a copy of the email message I received from Mr. Squibbs requesting the revised review. Exhibits D, F, and G were transmitted and/or received during the period between just prior to February 29, 2000 until October 17, 2000.

8. Attached as Exhibit H is a table comparing the elements of claims 19 and 36 with the corresponding conception identified in Exhibits A and B.

9. Exhibits A-G, which relate to the aforementioned conception of the claimed invention followed by diligence until a constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Actual dates of Exhibits A-B have been removed, but are prior to February 29, 2000. Exhibits D, F, and G, which relate to the aforementioned diligence until constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Further, company proprietary information has been removed from all

exhibits.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.



Date: 14th Nov 2006

Colin Andrew Low

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400
Telephone: 703-684-1111
Facsimile: 970-898-0640

RAN/

Docket No.: 30004635-2 (1509-226) **PATENT**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of	:	
HAWKES, RYCHARDE JEFFERY *et al.*	:	Confirmation No. 1484
U.S. Patent Application No. 09/977,501	:	Group Art Unit: 2143
Filed: October 16, 2001	:	Examiner: Jude Jean Gilles

For: CONTENT PROVIDER ENTITY FOR COMMUNICATION SESSION

Declaration Under 37 C.F.R. §1.131

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I, Robert Francis Squibbs, based on information and belief, do hereby declare that:

1. I am a patent attorney registered to practice in the U.K. and employed by Hewlett-Packard Company (HP), a wholly owned subsidiary of Hewlett-Packard Company.

2. In the course of my representation of HP, I was requested to prepare a patent application based on an invention disclosure submitted by Rycharde Jeffery Hawkes, Lawrence Wilcock, and Colin Andrew Low. After receipt of the disclosure and approval for patenting, preparation of the U.K. patent application was entered into our docketing system in order that preparation could be scheduled among other applications awaiting to be drafted and taken up in order as other application prosecution work allowed. In due course, I prepared a patent application which was filed in the U.K. Patent Office and which corresponds to U.K. Application No. 0025454.0 (the U.K. Application) filed on October 17, 2000. U.S. Patent Application having a serial number of 09/977,501 (the '501 application) was filed on October 16, 2001 and claims priority to the U.K. Application.

3. Prior to February 29, 2000, I received a disclosure of the invention in the form of a document entitled, "A Web Interaction System." Attached as Exhibit A is a copy of the disclosure document. Exhibit A describes a Web Interaction system created for CRM, E-commerce, and customer interaction applications and services.

4. Prior to February 29, 2000, I received additional disclosure of the invention in the form of a document outlining potential patent claims entitled, "Appendix 1 - Potential Patent Claims." Attached as Exhibit B is a copy of the document.

5. Attached as Exhibit C is a copy of an email message I received prior to February 29, 2000 from one of the inventors providing me a copy of Exhibits A and B.

6. During the period between just prior to February 29, 2000 until October 17, 2000, I worked with the inventors and drafted the U.K. application. During the afore-mentioned time period, I employed reasonable attorney diligence in preparing the application by working reasonably hard on the application during the continuous critical period. During the critical period, I maintained a reasonable backlog of unrelated cases which I took up in order and carried out expeditiously. During the afore-mentioned time period, I received another detailed document, attached as Exhibit D hereto, describing information related to the Exhibit B document. Attached Exhibit E is a screenshot of the Microsoft Word file properties dialog displaying the date of creation of the file corresponding to Exhibit D. Attached as Exhibit F is a copy of a version of the specification and drawings which I drafted and transmitted to the inventors requesting review thereof for the above-identified application. During the afore-mentioned time period, the inventors reviewed the application in order to provide comments and corrections to myself. During the afore-mentioned time period, I requested the inventors to review a revised version of the claims of the application prior to filing the application.

7. During the period between just prior to February 29, 2000 until October 17, 2000 and in accordance with HP standard application procedures, I proceeded to direct the filing of the U.K. application with the U.K. Patent Office on October 17, 2000. Attached as Exhibit G is a copy of an email message I transmitted to the inventors requesting the revised review.

8. Attached as Exhibit H is a table comparing the elements of claims 19 and 36 with the corresponding conception identified in Exhibits A and B.

9. Exhibits A-C, which relate to the aforementioned conception of the claimed invention followed by diligence until a constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Actual dates of Exhibits A-C have been removed, but are prior to February 29, 2000. Exhibits D, E, F, and G, which relate to the aforementioned

diligence until constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Further, company proprietary information has been removed from all exhibits.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

_____ Date: _____

Robert Francis Squibbs

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400
Telephone: 703-684-1111
Facsimile: 970-898-0640

RAN/

A Web Interaction System

Rycharde Hawkes, Colin Low, Lawrence Wilcock

Hewlett Packard Laboratories

Draft 1.0

Abstract

This document describes a Web Interaction system created for CRM, Ecommerce and customer interaction applications and services. It was designed and built by members of HPLabs, Bristol, in collaboration with the Customer Relation Management Organisation (CRMO).

This document was written primarily to document intellectual properties embodied in the system, but it provides a self-contained overview with a modest level of technical detail.

1 Contents

Exhibit A

2 Background

The Internet and the World Wide Web (WWW) have made it possible for enterprises to sell products and services by using the WWW to describe offers, using various means such as WWW forms or electronic mail to conduct transactions. This form of selling is based around the catalogue model that originated in the 19th. Century, where the WWW site substitutes for the paper catalogue, and the postal service is replaced by the modern online equivalent.

Many enterprises currently use the telephone to replace or augment the catalogue model. A customer can call the organisation and purchase goods and services interactively over the telephone. This has the advantage that a customer can interact directly with a Customer Service Representative (CSR), but has the disadvantage that the telephone is a non-visual medium.

The need to handle large numbers of customers simultaneously, and the concurrent need to manage a pool of CSRs, has led to the development of the *call centre*, and the development of specialised software control packages to determine how incoming customer calls are routed to CSRs.

It is possible to combine the catalogue model of WWW selling with the telephone call centre (and other communication channels) to produce what is often called *contact centre*. The contact centre is like a telephone call centre, but instead of CSRs handling only telephone calls, they may be expected to handle customer communications in a variety of formats: FAX, electronic mail, telephone and WWW are typical. A contact centre is characterised by multiple contact or communication channels, and a pool of CSRs who interact with customers to provide services, products or support. The contact centre provides the illusion of a single point of contact for customers on a regional, national or even international basis.

The online purchasing experience can be further enriched by taking ideas from the familiar mall or high street shopping experience, where the social element of viewing and purchasing goods and services becomes important. It is common for people to go shopping together, to inspect goods and offers together, and to discuss the pros and cons of a given item. Sales assistants can operate in a number of different modes: they can approach a customer who looks in need of help, they may work in a specific area such as cosmetics or cameras, they may work on an enquiry counter, they may provide demonstrations of appliances, or they may be generally available and circulate around different parts of a store. It is common for sales assistants to consult each other and share knowledge, and it is also common for senior sales staff to monitor the performance of junior sales staff as part of their training. For the sale of high-ticket items, there may be several parties on the buying side and several parties on the selling side.

Creating the on-line analogues of these very familiar practices requires a richer communication infrastructure than the telephone call centre or the Internet contact centre, an infrastructure characterised by the following features:

- Multiple consecutive communication methods. The customer may use several different methods to contact the same organisation on different occasions, such as electronic mail on day one, followed by a telephone call on day two.

- Multiple concurrent communication methods. The customer may use several different communication methods concurrently, something that is becoming possible using the inherent multimedia capabilities of the Internet. Several media types may be used concurrently within the same session, examples being audio (speech), video, graphics, WWW content and text chat.

- Multi-party communication sessions. There may be several customers and several CSRs in the same communication session.

- Session Modification. An ongoing session may be enlarged to include more people, or contracted as people leave. The attributes of a session or a session member may be modified. There will be privileged supervisory modes with enhanced session control and modification capabilities.

- Session Routing. A customer (or CSR, or supervisor) may select a session to join using a variety of descriptive attributes associated with a session.

- Security. The open nature of the Internet necessitates a robust approach to security to protect customer confidentiality.

To summarise, the praxis of selling goods and services has evolved in recent history to incorporate appropriate technology. The face-to-face direct sell, usually based in specialised retail premises, has been extended to include catalogue selling (based around postal mail) and call centres (based around the telephone). The growing success of the Internet as a medium for selling and buying products and services raises the question of how to employ the underlying capabilities of the Internet to improve the customer experience. The inventions described in this document make possible a range of novel and innovative systems for real-time multimedia interactions between customers and CSRs, with the goal of making on-line purchasing a productive and enjoyable experience.

Section 3 provides an informal description of two terms used throughout this document, the *communication session* and the *transport session*. Several customer and CSR interaction scenarios are described in section 4, and these scenarios provide the justification for the architecture. In section **Error! Reference source not found.** the idea of a CSR is broadened to include the automaton, "bot" or *virtual representative*. Section 6 provides an overview of how the system is instantiated in today's Internet environment, and outlines the interactions between different parts of the system. Section 0 describes the architecture.

3 The Communication Session

3.1 The Communication Session

The communication session is a core concept in this web interaction system. A formal definition is given in section 7.4. This section provides an informal description to assist in reading the overview sections of this document.

A communication session corresponds to a group of people in a meeting room. Two or more people can share a room to discuss topics of mutual interest. This familiar model has the following characteristics:

- It is private. Communications are restricted to those people in the room.

- It is shared. Each person is potentially a full participant.

- There are many modes of communication. People can talk, make notes, draw on a whiteboard, show slides, watch a video.

- People can come and go during the meeting.

- Some people may be restricted as to which modes of communication they can use in the meeting. A blind person can talk and listen, but not see a slide presentation.

- One or more people may have privileges to moderate the meeting (e.g. a chairperson).

- There needs to be some method for discussing how to locate the meeting.

Communication technology can be used to provide people who are geographically remote with the illusion that they are sharing a meeting space. For social reasons it is useful to preserve as many of the relevant attributes of real meetings as possible.

A *communication session* is an abstraction that corresponds to a real meeting. We use the term extensively throughout this document. This abstraction can be realized using computer and Internet technology, and is familiar to those skilled in the art. The specific application of this general idea to the problem of customer support, customer interaction and customer collaboration is novel and is described in detail in the Architecture section of this document, and in still more detail in our claims.

3.2 The Transport Session

The communication session defines a meeting space and participants. The *transport session* defines what is communicated, who to, and how. Each communication session requires a transport session.

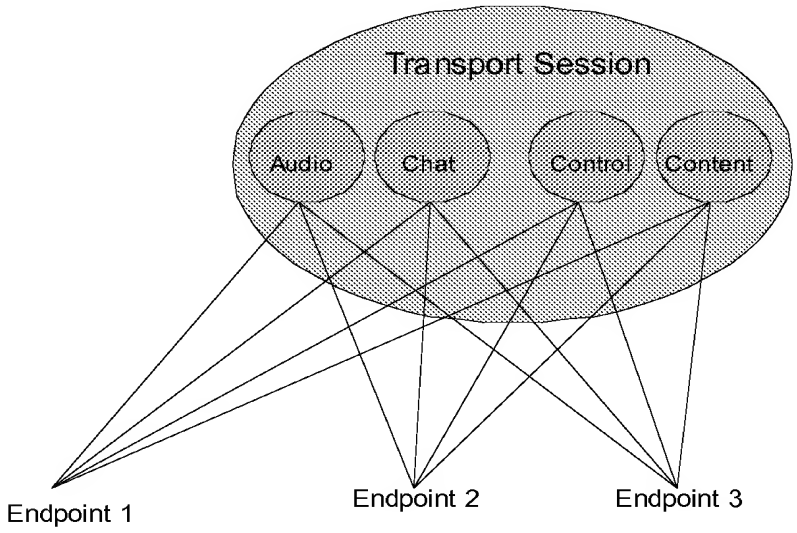


Figure 1: Media Channels in a Transport Session

The transport session is structured into one or more media channels. Figure 1 above shows four media channels. Each media channel is used to send and receive media content of a particular type. In a typical application, media content sent to a channel is received by all channel members, a subset of transport session members. In our embodiment there is a one-one correspondence between communication sessions and transport sessions, and between participants in a communication session, and members of a transport session.

4 Session Scenarios

4.1 Background

In the scenarios described below the following terms are used:

- Text Chat. Each member of a session can type lines of text into a chat GUI at any time. These are sent to other session members in real time (i.e. with a delay on a few seconds) and displayed in a chat window as an interleaving along with the name of the person sending the text.

- Page Push. The page corresponding to a WWW URL is displayed in a reserved browser window of each session member. In a "Follow-Me Tour", clicking on a hyperlink on the page in the Page Push window results in all session members following that link in synchrony. Page Push, and its variants, is a way for session members to share WWW content.

- Callback. A session member can be called-back at their telephone number. This feature is common in telephony call centres, and a telephony call centre will have dedicated hardware for terminating and routing incoming telephone calls to CSRs. This hardware will usually have the ability to originate calls, making it possible to set up a dialback call between a CSR and a customer. This capability is an example of hybridisation between existing call centres, which are oriented around telephony, and the next generation of Internet Relationship Management centres which use Internet technology for communication with a customer. In our preferred embodiment callback is implemented only in one-one customer-CSR sessions.

- Deferred Callback. A customer is called back at a nominated time.

4.2 Online Help

A customer is browsing WWW pages belonging to an enterprise and wants to talk to a CSR in that enterprise. The page will have some kind of "Help" button hyperlink which the customer clicks on.

The customer browser then progresses through a WWW dialog which makes it possible for the customer to identify themselves by submitting a small number of personal details (e.g. name, customer reference, email address etc). The customer browser then launches graphical user interfaces (GUI) for each of the media types used in a session. This will typically be page push, and text chat or voice chat. An available CSR is discovered, joins the session, and the customer and CSR can then begin to discuss the issue that caused the customer to request help.

The session can be extended by inviting-in additional CSRs, and the "call" can be transferred to another CSR. The session can also be extended to include additional customers.

4.3 Online Help with Dialback

As in the "Online Help" service, but the customer provides a Public Switched Telephone Network (PSTN) number so that the CSR (in fact, the telephone callback hardware referred to above) can dial back to the customer, so creating a voice channel in addition to the other communication channel. The PSTN is normally used for voice traffic because it provides a higher quality channel for voice communication at the current time.

4.4 Online Help with Deferred Dialback

As in "Online Help with Dialback", the difference being that the customer wants to talk to a CSR at some future specified time. This is useful when all CSRs are allocated, and the customer wants to reserve a callback at a convenient time.

The customer goes through the initial dialog, provides personal details including a telephone number, and is provided with the URL of a page to return to at a later time.

When the telephone callback occurs, the customer goes to the URL provided, and information about them previously stored in the WWW browser is used to identify the customer and callback. The session GUI is launched by the customer's WWW browser and connects to the correct collaboration session. The CSR is also invited into the same session.

Figure 2: Shop with Friends

4.5 Deferred Dialback

The customer uses the initial WWW dialog to select a telephone callback at a specified time. No collaboration session is created, and the Internet is not used as a communication technology.

4.6 Shop with Friends

This scenario assumes that two or more friends want to browse and make purchases online. They would want to communicate with each other using text or audio chat, and see the same WWW pages in a straightforward "Follow Me" tour as described above. In Figure 2, three friends are viewing the same page together, discussing its content, and they have invited a CSR to answer some questions. From the customer perspective this scenario is identical to the "Online Help" scenario, with the exception that the session members are customers and not a mix of customers and CSRs.

The first friend initiates the session using a WWW dialogue to submit a small number of personal details, and is given a session identifier/password which needs to be communicated to all other friends by some other means (e.g. email, instant messaging). Each friend would go to a WWW page, type in the session identifier/password, and would become a member of the session, and so would be able to see the same WWW page as other session members (because

of Page Push/Follow Me) and would be able to engage in text or audio chat with other session members.

4.7 Page as Place

The "Shop with Friends" scenario uses a fixed multi-party session, and a succession of WWW pages "flow throw" the session using follow-me page push. Session participants effectively wander around the WWW, the session maintaining its coherence as it travels.

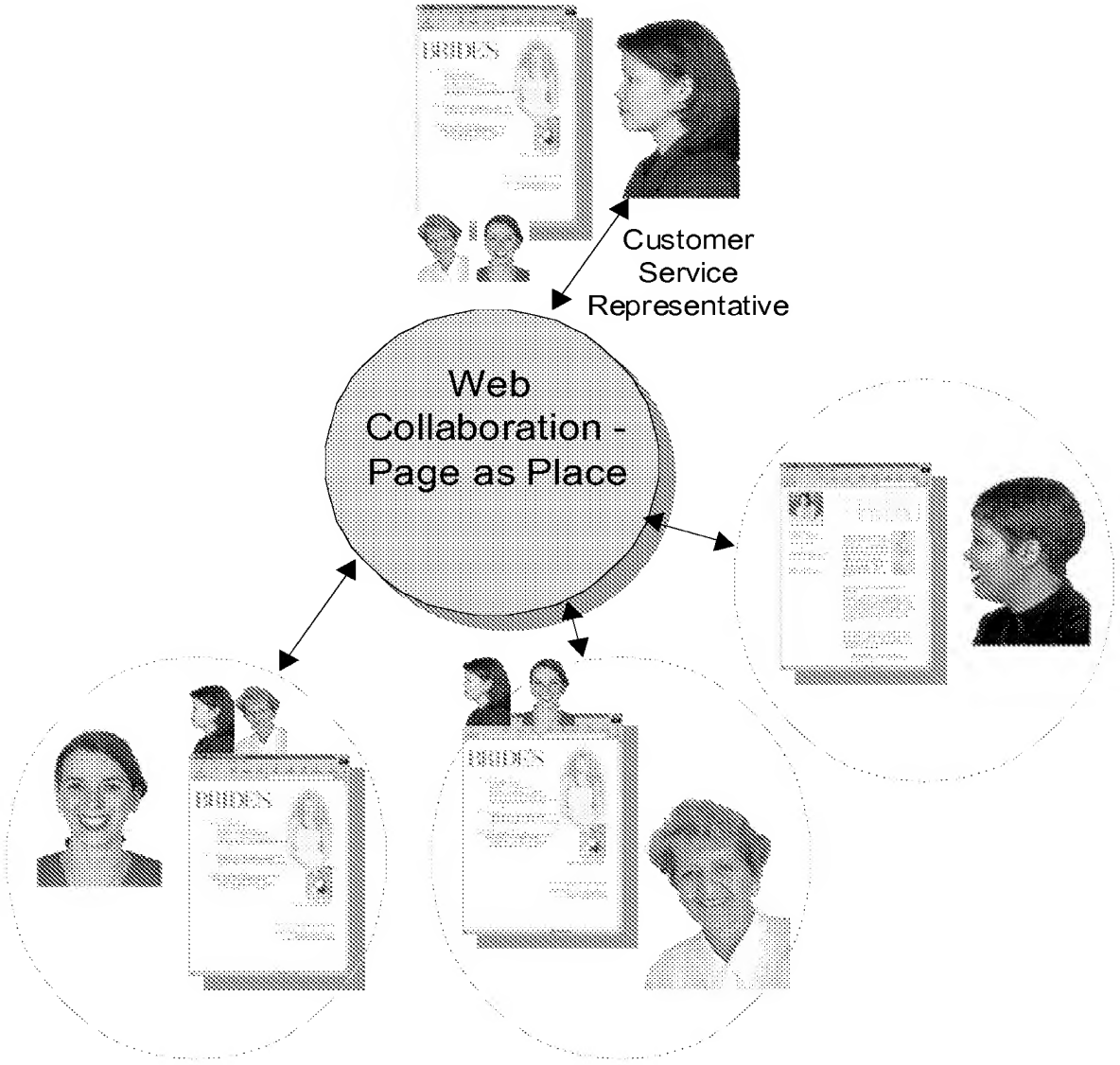


Figure 3: The "Page as Place" Service

An alternative is the "Page as Place" scenario, where a communication session is immutably associated with the page. In this scenario, as customers move from page to page, they move from session to session. Figure 3 shows four women looking at two different pages. The woman on the right is in a session by herself. The two women at the bottom are in the same communication session, and have been joined by a CSR who is monitoring activity on that page.

The advantage of this scenario is serendipity: it corresponds closely to what happens when a person wanders around a mall, meeting a different set of people in each shop. Wandering into a page showing lawnmowers, one can choose to see whether anyone else is also looking at lawnmowers, and engage them in conversation. One might see a CSR just "standing around" on the page, or one could listen in on what a CSR was telling another customer (these customer interaction services blend into each other, and the "page-as-place" scenario and the "customer presentation" scenario have a great deal in common.

There is a great deal that can be done with this simple concept. Instead of thinking of a WWW site as a catalogue, it can be organised like a department store or a mall into a set of places – the perfume department, the coffeee shop, ceramics, cooking an so on. Instead of a customer having to decide whether their query is important enough to justify contacting a CSR, they can see CSRs "standing around" when they move from page to page. It is not considered an imposition to approach an idle sales assistant in a shop even for the most trivial of queries, because we know that is what they are there for.

5 Virtual Representatives & Bots

5.1 Introduction

Users of the public telephone system will be familiar with automata in the form of the Interactive Voice Response (IVR) system. These are use in applications such as voicemail, call centres, and cinema listings, and consists of a recorded or synthetic human voice that offers a menu of choices, and the customer interacts by using telephone keypad tones. More sophisticated automata are finding their way into telephone directory enquiries, into business premises as telephone receptionists, and as "virtual personal assistants" for business telephone users.

A common term used for human automata is "bot", a contraction of "robot", with the understanding that a bot is a software entity that interacts like a real human being and is capable of providing simple services. The term appears to have originated in multiplayer game environments (MUDs, MUCKs, MOOs) where such bots are a common feature.

A real, human CSR interacts in a communication session using a desktop GUI application which provides media GUIs for each media type. In our embodiment each media GUI, and the overall session control and interaction logic, is divided into two parts according to a method understood by those skilled in the art. This separation is often called "model/view" or "semantic/presentation", and refers to the split of a GUI application into a part that contains the core logic and data (the model), and a part that presents this to the user (the view).

By splitting the application into two parts in this way, and by using standard object-oriented programming techniques (e.g. Java Beans) it is possible for a software automaton to interact with a session in exactly the same way as a human being, with the difference that a human being interacts via the view components, and the automaton interacts via the underlying model components.

To give an example, suppose the chat component receives a "new chat text" event, and delivers a line of text chat to be displayed. For a human being this text would be passed to the view component for display in the chat window, while for an automaton the text would be passed directly into the natural language parser for the automaton.

The outcome is that an automaton can do anything in a communication session that a human being can do. The challenge is to improve the intelligence of software automata of this kind to the point that they provide economically useful tasks.

This section introduces a number of bots which are designed to participate in communication sessions as they are described in this document.

5.2 Marketing on Hold

When a customer requests help from a CSR, there may be a period of time before a CSR becomes available. During this time the customer just has to wait. In telephone call centres it is common to play music.

It is possible for a "Marketing on Hold" bot to join the customer in the session in the period before a real CSR is available. The bot would use the media channels in the session to interact with the customer, probably by providing special (possibly personalized offers) or entertainment. There are many other possibilities.

5.3 Content Monitoring

In telephony-based call centres, supervisors monitor calls in order to ensure a certain level of service and to curtail the negative aspects of the relationship that is often formed between CSR and customer (especially in business-to-business). As a CSR becomes familiar with their customers, there is a tendency for them to become a little more "chatty" than the situation demands. The supervisor also monitors calls to ensure that the CSR is professional in their interaction with the customers and that the customers are neither party receives abuse from the other.

A large part of this job can be achieved by wandering around the CSRs and listening in on the conversations (at least from the agent's side). When other forms of interaction are introduced, this no longer becomes an effective way to monitor the calls. Specifically, with chat it would be easier for a CSR to unnecessarily extend the duration of the call or use profanity without the supervisor being immediately aware.

A content monitoring Bot could be used to help the supervisor in their job. This Bot could perform a number of functions:

1. Detect the presence of profanity or abusive terms and phrases.
2. Detect the pushing of URLs that may lead to undesirable material.
3. Measure the response rate of the agent, i.e. some measure of time between receiving content on a channel and responding to it.

For the first two functions, the Bot would also identify whether the CSR or the customer was responsible for the indiscretion. The Bot could then notify the supervisor's desktop, providing a copy of the transcript so far and the suspicious content. The supervisor would then decide whether further action is necessary, e.g. joining the session, terminating it, etc., or to ignore the report. The Bot may also communicate with the Transcript Bot (see section 5.4) to ensure that this transcript is saved for further examination. Indeed the functions of the Content Monitoring Bot and the Transcript Bot may be merged into one process.

The third function is a useful metric to be added to the existing statistics collected per call, e.g. call duration. A long call duration with a low response rate might indicate an overloaded agent (two many simultaneous calls perhaps) and a long call duration with a high response rate might indicate an overly "chatty" agent.

5.4 Transcript Bot

When the technology described in this document is used for customer support, the transcripts between customers and agents are valuable because they provide a record of a problem and its solution. These records can be analysed off-line to provide better responses to future customer enquiries. A transcript can also be appended to a customer record as part of an ongoing dialog, a filing practice that is common in paper systems.

The transcript bot would join a communication session and record a transcript for the session. It would provide additional interfaces to other services for storing or replaying the transcript.

5.5 Virtual Representative

A *virtual representative* would provide customer service when no human operator is available. This is useful for organizations that need to maintain a 24 hour presence on the WWW but are unable to justify the expense of providing a 24 hour rota of human operatives.

6 The Web Collaboration Server – an Embodiment

6.1 Physical Overview

The Web Collaboration Service is a specific embodiment of the architecture outlined in this document. Figure 4 shows an example of how the service might be embodied on a set of computer servers running on Local Area Networks (LANs) connected to the public Internet.

We assume a customer network consisting of one or more customers connected to the public Internet. It is common in many organisations to use a *firewall* to isolate the private internal network from the public Internet for security reasons. The customer could be a domestic consumer connected via an Internet Service Provider (ISP) or it could be an employee of an organisation with a very high level of internal security, such as a bank or a hospital. We assume that the customer is connected through the customer premises firewall using the standard Internet TCP/IP protocol, and in particular that the customer has World Wide Web access using the standard Hypertext Transfer Protocol (HTTP).

We assume a similar environment for the Web Collaboration Service. It is hosted by a collection of servers interconnected by a LAN, which is also connected to the public Internet through a firewall, using the standard TCP/IP protocol. This kind of physical embodiment is common when accessing services using the WWW and would be recognised by anyone skilled in the art.

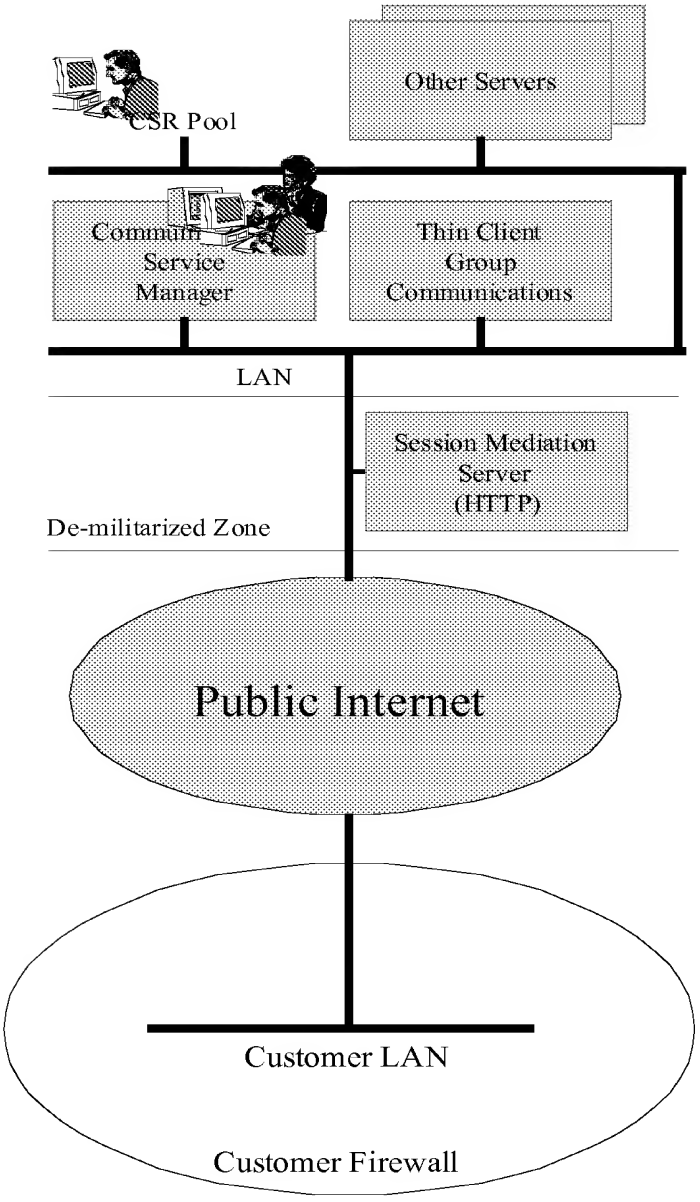


Figure 4: Physical Embodiment

It is a principle of distributed systems design that services should be designed to permit maximum flexibility in the physical distribution of these services on actual physical computers. Consequently, the following names and quantities of servers hosting the Web Collaboration Service are shown for the purposes of illustration and clarity, and should not be read as determining a unique physical instantiation of the architecture.

As we assume access to the Web Collaboration Service using the public Internet, there needs to be some way for customers to interact with the system, primarily to initiate collaboration sessions. Referring to Figure 5, the Session Mediation Server is a WWW (HTTP) server which hosts an extensible set of WWW pages used to initiate communication sessions. In our embodiment this server is an Apache Web Server (see www.apache.org) running the Jserv (java.apache.org) Java Servlet environment. Java Servlets (see javasoft.sun.com) running on the server are used to parse WWW forms, validate customer inputs, and these interact with

the Communication Service Manager, which oversees a collection of concurrent collaboration sessions.

Figure 5: Interaction Scenario

The Service Mediation Server has to be accessible to customers across the Internet, and so it would usually sit in a special part of the firewall called the "De-Militarized Zone" (DMZ), which uses security techniques understood by those skilled in the art to permit access to servers in the DMZ while denying access to servers behind the DMZ. The main part of the collaboration service sits behind the DMZ where it is inaccessible to customers.

There are two main components behind the DMZ. These are the Communication Session Manager (CSM), which is used to create, manage and terminate *communication sessions*, and the Thin Client Group Communication Server (TCGC) which hosts *transport sessions*. The CSM communicates with various other servers such as corporate databases and call centre suites using techniques which are outside of the scope of this document, but can be encapsulated in a set of libraries for linking-in external applications. Finally, there is a pool of Customer Service Representatives (CSRs) who are available to take part in Communication

Sessions with customers, in the same way as a CSR is automatically allocated to a customer in a conventional call centre.

Several computer communication protocols are used in this embodiment. All are layered on top of the standard TCP/IP protocol, and all would be familiar to someone skilled in the art. These are:

- Hypertext Transfer Protocol (RFC1945). This is the protocol used by WWW browsers to access WWW servers for the purpose of providing the WWW service. This protocol is the lowest-common-denominator means for customers to interact with the collaboration service, and it can be transported securely across corporate firewalls.

- Java Remote Method Invocation (RMI) (see javasoft.sun.com) is a protocol used to invoke methods on remote objects in distributed systems using techniques widely understood in distributed systems. It is used (for example) between the Service Mediation Server and the Communication Session Manager for invoking session initiation operations provided by the CSM.

- Java Message Service (JMS – see javasoft.sun.com) is a public specification for inter-computer messages which has been widely implemented to disguise various proprietary protocols, and is a convenient way to specify and implement interactions between the CSM and other back-end servers, such as databases, call centre suites, and CSR desktop applications.

6.2 Physical Interaction Scenario

The following scenario is intended to provide a coarse-grain view of interactions between physical components in the system. It is an example of a basic and familiar service where a customer browsing the WWW wants to contact a CSR to ask questions. The service takes information about the customer, creates a communication session, invites the customer into the session, locates an available CSR, invites the CSR into the session, and once both parties are in the session, they can begin to interact using various media. In our current embodiment they would interact using text chat and WWW page content.

The detailed interactions are as follows (see Figure 5):

1. The customer is browsing an enterprise WWW site and wants to talk to a CSR about some issue. These WWW pages are outside of the scope of the system being described. The customer finds a "help" button on the WWW page and clicks on it. This button is a hyperlink to a WWW page on the Session Mediation Server.

2. The customer goes through various hypertext/WWW dialogues on the Session Mediation Server (SMS). This involves selecting various communication options, and the customer supplies a small amount of personal information. The servlet running on the server can also extract information about the customer in the form of "cookies" and other information from the HTTP header request – these are normal activities on a WWW server and would be understood by a WWW system designer skilled in the art.

3. The SMS condenses all the information about the customer in the form of a Java object and communicates it to the CSM using the Java RMI protocol. There may be additional communication between the SMS and external databases containing customer information, so that the information presented to the customer can be personalised according to the past history of interactions.

4. The CSM creates an empty communication session for the customer and communicates with the TCGC server to create a transport session. Information about the transport session is returned to the SMS, which returns the customer interface to

the session as part of an HTTP response (i.e. a WWW page containing active content such as Java or Javascript). In this way the customer loads a WWW page which contains information about the selected transport session.

5. The customer desktop software (typically active content running in a WWW browser) joins the transport session. This involves creating media channels within the transport session for each of the media types in the session.
6. The CSM interacts with an external service (e.g. part of a call centre suite) to select an available CSR
7. The CSR desktop software receives an invitation to join the transport session. This invitation contains customer information.
8. If the CSR accepts, the desktop software joins the transport session.

At this point, both the customer and CSR can exchange information using media channels created as elements of the transport session. The CSM monitors the status of the transport session, and when either customer or CSR leave, it tears down both the communication and transport session.

This physical infrastructure is capable of supporting many service scenarios. The SMS provides mediation between the world of the public Internet and its protocols (i.e. HTTP) and various back-end servers, such as the CSM, corporate databases, e-commerce systems etc. Because of security hazards its functionality will often sit on a ring-fenced LAN connected to the public Internet (the DMZ), and its communication with back-end servers is strictly limited to specific forms. Because the SMS generates the user interface for the customer (in the form of WWW pages) it acts as a proxy for the customer user interface. In the Lite customer desktop, all the logic needed to join and be a member of a transport session runs in the SMS as a collection of Java servlets and Java packages.

The TCGC server provides transport sessions. Each transport session can contain many media channels, which provide multi-party delivery for each media stream. Although only one TCGC server is shown, there can be many. Media delivery is a processor intensive activity and is proportional to the rate at which media is delivered to each channel. In the case of audio or video the data rate associated with a channel can be very large.

Transport sessions are created and managed by the CSM. It becomes involved only when the state of a communication session changes – for example, when the membership of the session changes. The CSM manages a pool of concurrent communication sessions and the associated transport sessions. It also contains the logic for each type of communication service, several of which are described in this document.

It should be noted that the scenario described here assumes a physical organisation that is not unique. Labels such as CSM, SMS and TCGC denote *convenient packages of functionality* that can be distributed over many computer servers or co-located on one single computer. As packages of functionality the labels specify no precise function – they are placeholders for functionality of a specific type. Part of that functionality is generic, such as *transport sessions* and *communication sessions*, and is described elsewhere in this document. Part of the functionality is specific to a service and is usually described in detail under a (claims for invention – term.?).

6.3 CSR Desktop

6.3.1 Introduction

The CSR Desktop (Figure 6) is the agent's sole point of interaction with web channel calls but may be used in conjunction with other channels, e.g. telephony, to offer richer service,

e.g. voice and page push. A CSR is associated with one or more *campaigns* which are a way of breaking down a customer service problem space into smaller logical areas.

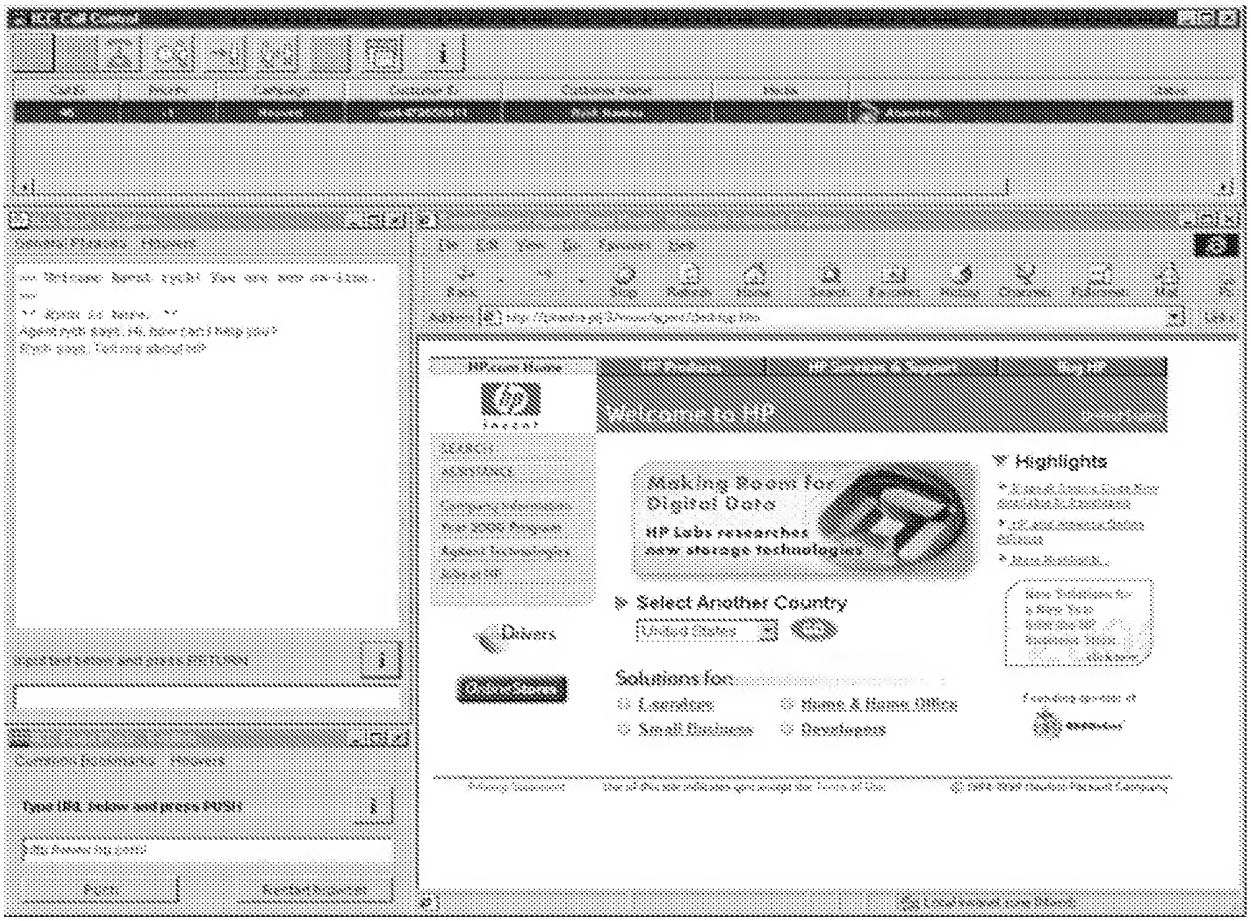


Figure 6. CSR desktop.

6.3.2 Multi-tasking

In telephony, interaction between customer and agent is swift and if the agent wishes to split their attention between two or more calls, they must put one of the calls on hold, thus breaking the appearance of dedicated service. With the web channel, interaction can be less intense, e.g. if the customer is not familiar with a keyboard, and there is opportunity for an agent to multi-task between a number of calls.

To support the illusion from the customer's perspective that the agent is giving them dedicated service requires a GUI that enables the agent to manage the information and media associated with each call quickly and effectively. Figure 7 shows an example GUI that can be used by the CSR to receive incoming calls and managed calls that they are already dealing with. Each call appears as a row in a table containing relevant information such as customer name, customer ID, the campaign this call belongs to and the status of the call. To answer the call, the CSR selects the row containing the call details and presses the appropriate button.

Figure 7. New call presented to CSR.

6.3.3 Media GUIs

The CSR desktop has been configured to expect calls with associated sessions that contain a certain set of media types. The desktop creates GUI elements to provide visual interfaces to the media clients that must be created for that call. Figure 8 shows a GUI element that interfaces with media clients handling content for a text-chat media type. In this case the GUI shows that no call has been selected.

Figure 8. Media GUI for the media type 'chat' (inactive).

When the call is answered, the CSR desktop creates a media client for each media type associated with the session for the call. (The media description for the call is passed to the CSR desktop as part of the call presentation). The chat GUI now looks similar to Figure 9.

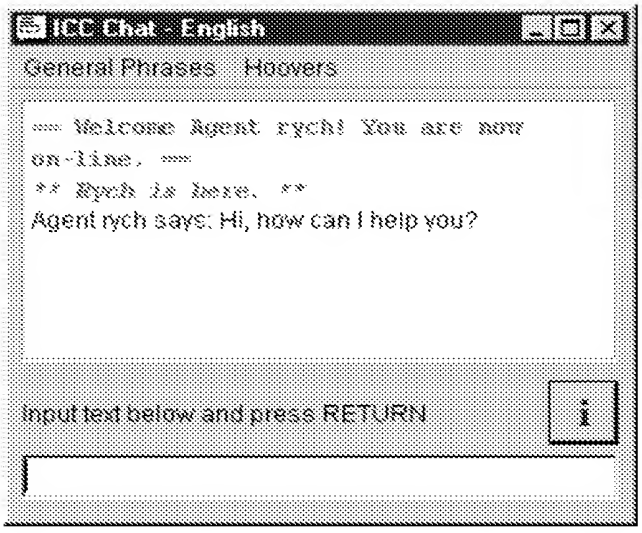


Figure 9. Chat media GUI for selected call.

Another useful media client is for pushing pages to web browsers of other clients in the session. Associated with the media GUI shown in Figure 10 is a web browser which displays the URL entered into the media GUI. As other clients push pages, the media GUI displays the URL and then tells the web browser to display the page.

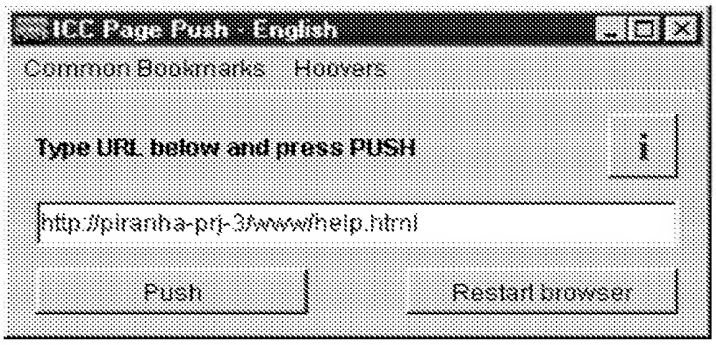


Figure 10. Page push media GUI for selected call.

6.3.4 Call Selection

When a new call is presented to the CSR, it appears in the top row of the table. If the CSR answers this call as well, then the call control display would look like Figure 11. Only one call can be selected at one time. When a call is selected, the respective media GUIs are updated to show the channel content relevant to that call. This approach minimises the amount of desktop space required for the CSR desktop application and reduces the amount of information a CSR must process at any one time.

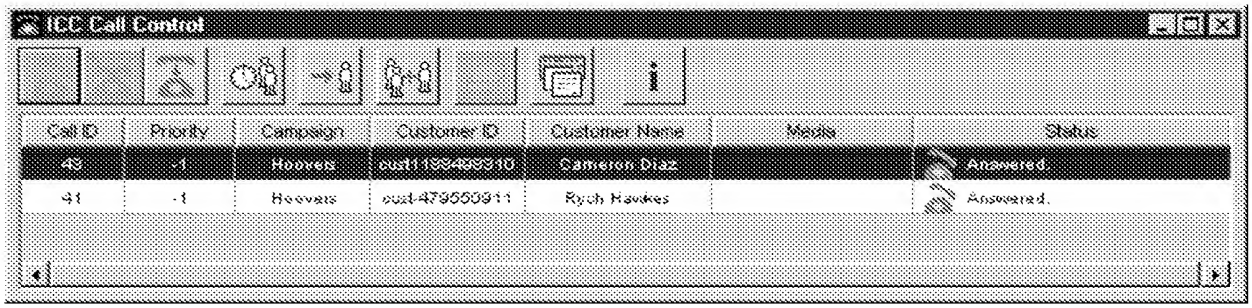


Figure 11. Second call answered by CSR.

While dealing with one call, some new content may appear on the media channels associated with one of the other calls. When this occurs, an icon representing the media type is displayed in the call table's media column. This simple mechanism enables a CSR to concentrate on interacting in one session in the safe knowledge that they are missing input from another customer. As soon as the media icon(s) appear, they can select that call and check what has happened, making a response as necessary (when a call is selected, all media icons are cleared from the call's display). Using this process, a skilled CSR agent should be able to handle a number of calls simultaneously.

Not all calls have the same media types associated with them. When this occurs, the media GUIs reflect this fact, e.g. Figure 12 shows what happens when there is no chat media channel for the selected call.

Figure 12. Chat media GUI showing that no media channel is available for the selected call.

6.3.5 Multi-call Control

When the customer or CSR drops the call, the call details do not disappear immediately. Instead, the call table entry remains (Figure 13) and the media GUIs are left in a state where the CSR can still peruse the content generated when interacting with the customer. When the CSR is happy that they have captured all the information they need, the call can be removed from the call table and the media content is lost forever.

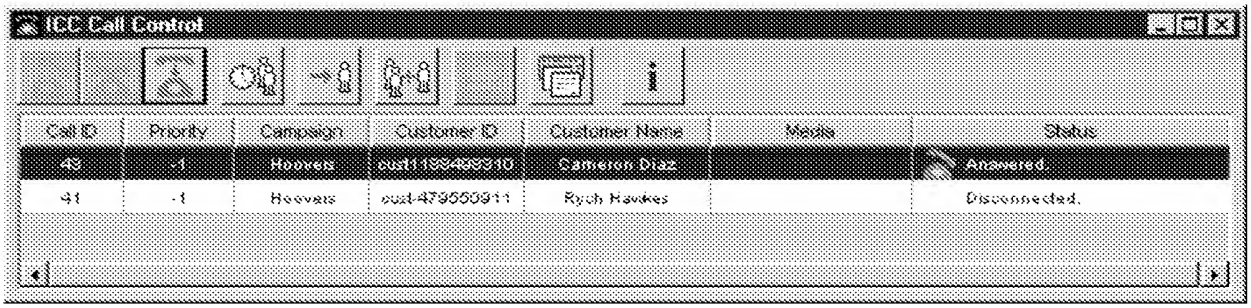


Figure 13. Call control after a call has been dropped.

6.3.6 Scripts

CSRs often have to deal with the same types of enquiries over and over again. To extract the most relevant information as efficiently and as quickly as possible, a CSR will often ask a series of predefined questions. These are commonly placed into scripts which are either followed from top to bottom, or sampled as necessary.

When using telephony this is sufficient. However, when presented with a medium of interaction such as text-based chat, the CSR is required to type repetitive phrases and questions in addition to the usual conversational pleasantries.

The menu bar in Figure 9 shows two scripts that are represented by pull-down menus. The first cascade menu contains commonly used phrases such as "Hi, how can I help you?", and so on. The second cascade menu contains the questions that are specific for the campaign related to the call. As an agent flicks from one call to another, the correct campaign script is automatically displayed in the media GUI.

The cascade menus for the page push media GUI in Figure 10 contains URLs in much the same way as browser provide a bookmark facility. These URLs are again ordered for general campaign-independent use and campaign-specific relevance. Selecting one of these URLs will push the page the session.

Using text chat and page push to show a customer around a product, for example, with this interface would require the agent to first say something (or choose an entry from a script menu) and then select the appropriate URL. It would be preferable if selecting a single script entry displayed a text message and sent the URL to the page push media client automatically[1].

This concept can be extended to all media clients, whereby a multimedia presentation is effectively scored by the selection of script items.

6.3.7 Languages

In addition to being assigned to one or more campaigns, an agent may also be multi-lingual and capable of dealing with enquiries in multiple languages. The country and language (together making a *locale*) of the customer is passed to the CSR desktop as part of the call presentation and enables the media GUIs to change their output and input to display content created in a different locale and create content that will appear correctly on the customer's desktop, respectively.

In the case of the chat media GUI, this results in the menu bar displaying script names and their contents in the appropriate language. If the scripts in the appropriate language have not been previously loaded scripts then they are done so upon demand. This allows the allocation of multi-lingual agents to change at run-time as well.

---

[1] Support for this feature has been built into the product but not used.

6.3.8 Call Operations

The basic functions available to a CSR that can be performed on a call are:

6.3.8.1 Answer

Accepts the selected call and connect to the session.

6.3.8.2 Decline

Refuse the selected call such that another agent will be selected to take it.

6.3.8.3 Drop

Used when CSR has finished dealing with the customer.

6.3.8.4 Transfer to CSR

Transfer the call to another specified CSR. If the receiving CSR accepts the call the desktop waits until all the receiving desktop's media clients are connected before automatically disconnecting their own for that call.

6.3.8.5 Conference to CSR

Sometimes, a CSR may wish to conference in another CSR, for example one with more knowledge on a certain matter. If the receiving CSR accepts the call, then the session is extended to include that CSR's media clients. As long as there is one CSR dealing with the call, the communication session will remain open.

6.4 Customer Desktop

The Customer Desktop (CD) proceeds through a number of states during its lifetime. When it is launched it is in the *connecting* state, at which time a connection to server in the call centre is established. When the server accepts a connection it starts the process to select an agent to take the call (*routing*) and creates the communications session. The CD creates the appropriate media clients and connects them to the session.

The attributes used to select an agent are primarily based on campaign and language although others may be used in addition. When the routing process starts a message is sent to the CD indicating this and provides an estimate of how long it will be before an agent becomes available and the customer's position in the wait queue. Periodic progress messages are transmitted updating this status information. When an agent has been selected a message is sent to the CD indicating the name and other details of the agent that will be dealing with the call. These last two steps may occur again if the chosen agent declines the call. When an agent accepts the call the CD enters the *connected* state and the agent joins the session to find the customer waiting for them.

The agent may transfer the call to another agent. If this happens the CD is informed and the customer informed so that they do no anticipate any further interaction until the call has been successfully transferred.

At any point during these stages, the customer or agent can disconnect putting the CD into the *disconnected* state. At this time all resources on the CD, CSR desktop and server-side relating to that call are released.

However, as with the CSR desktop, the CD preserves the relevant media content, e.g. the chat transcript, so the customer can review the information they gathered after the agent has disconnected.

6.5 Customer Desktop

The Customer Desktop is composed of two layers: one providing service-level logic and functionality, the other implementing the Graphical User Interface (GUI). Our embodiment provides to functionally similar desktops with different implementations, the Applet desktop and the Lite desktop.

6.5.1 Applet Customer Desktop

Customer desktops run in a web browser. The Applet Customer Desktop (ACD) uses the browser's Java Virtual Machine to offer a richer interface to the call centre than is available using the Lite Customer Desktop (see below).

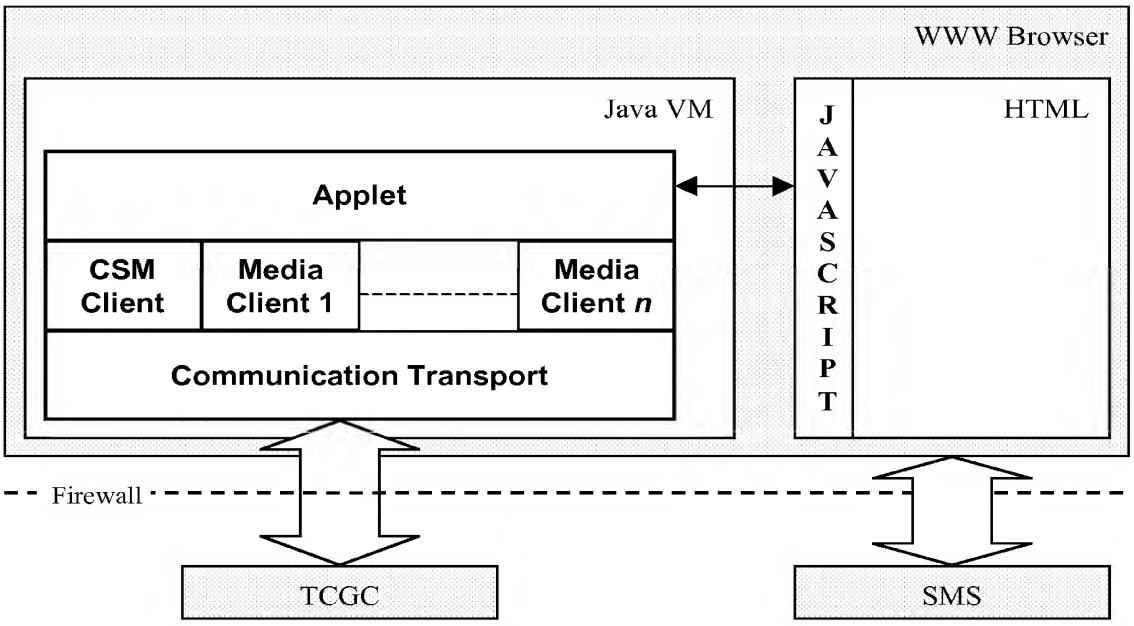


Figure 14. Functional structure of the applet desktop.

The applet customer desktop is launched when the SMS serves an HTML page containing the applet to be opened in the customer's web browser. The parameters for the HTML applet tag specify session information needed to initialise the desktop and connect to the CSM and TCGC, including:

- Customer identifier.
- Nickname (from the form filled in by the customer prior to desktop launch).
- Language the customer requested to receive service in.
- Address of the communication session associated with the call.
- The response to be given when challenged by the TCGC upon joining the session.
- The media description for the given session.

Figure 15. Functional structure of the applet customer desktop.The first action performed by the applet is to interpret the media description and create a media client for every media type contained therein. All the media clients are connected to the communication session with the given address and use the given response when challenged. A single CSM client is created and a connection with the CSM established. The purpose of the CSM client is to

communicate changes in desktop state, e.g. disconnection, to the CSM (and hence the CSR) and react to remote events, e.g. call transfer. Unlike the CSR desktop, all interaction with the CSM is through the communication session.

The ACD's GUI, is initialised so that input and output are suitable for the specified language and media GUIs created for the respective media clients. In this way, the user interface only contains those GUI elements that are strictly necessary for a given call.

The customer desktop progresses through a number of states during its lifetime (see section 6.4). The ACD uses an interface defined in Javascript to perform operations on and reflect certain events into the HTML document containing the applet. These events can be used for synchronising web content with the desktop. Just as the applet uses Javascript to access the HTML document, so Javascript can be used by the document to access the external interface of the ACD, e.g. get the current state of the desktop.

For example, in this embodiment, the initialisation event handler opens the browser window to be used to displaying the pages pushed by the page push media client. Upon receipt of a new URL from the page push channel, the page push media GUI invokes a Javascript method to display the URL in the page push window.

6.5.2 Lite Customer Desktop

The Lite Customer Desktop (LCD) uses HTML and Javascript in the web browser and locates the Java session functionality in the SMS. When the ACD is launched, the client establishes the connection with the TCGC. When the LCD is launched, a desktop proxy process is created in the SMS that connects to the TCGC and interacts with the CSM. The LCD forwards any user input, e.g. chat message, page to push, etc., to that proxy and polls it using an HTTP request for client updates, e.g. change in desktop state, new chat messages, page to display, etc.

In the current embodiment, the nature of the supported media clients is such that an almost identical user interface is presented to the user regardless of whether the LCD or ACD is used.

The LCD is much smaller than the ACD since there is no Java code to download, quicker to initialise but a little slower and less scaleable due to the polling.

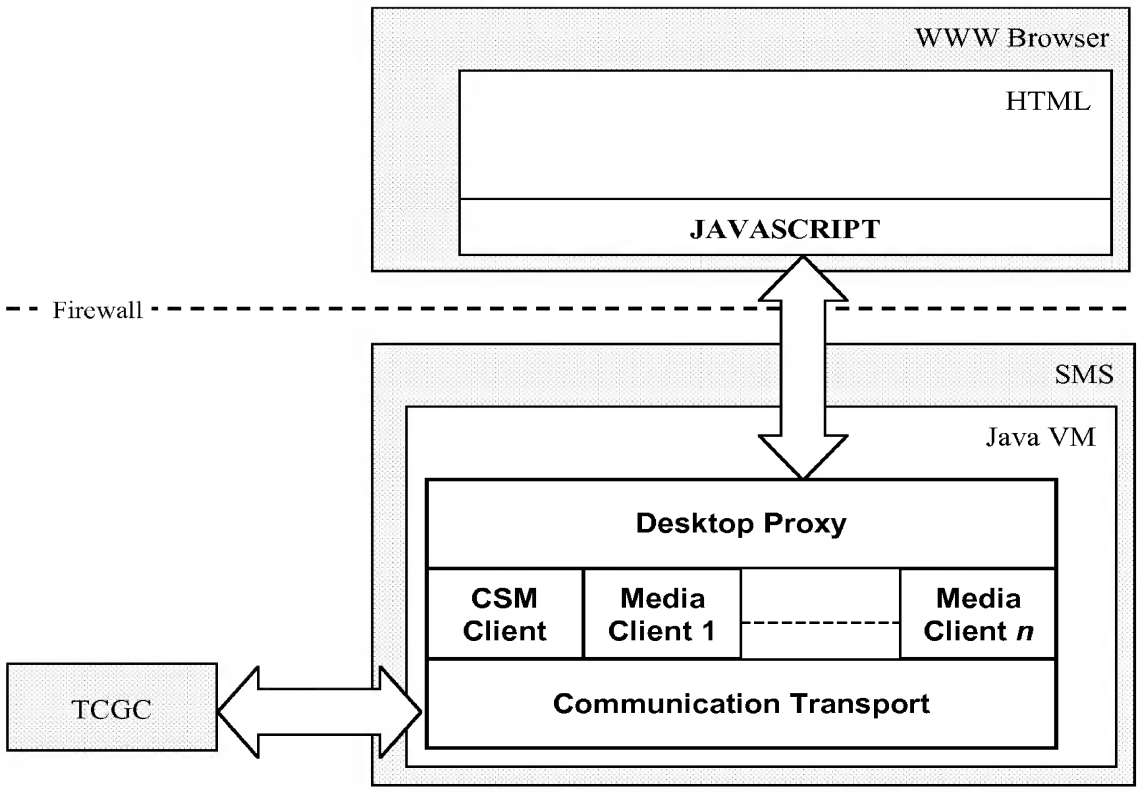


Figure 16. Functional structure of the lite customer desktop.

7 Architecture

The architecture describes the core building blocks of the system. It is does not describe how the building blocks are organized to form a working embodiment of the architecture.

An issue with describing modern distributed systems is that an architecture can give very little feeling for how a working embodiment might be organized in terms of the physical distribution of components over a number of computer systems. There are also many physical embodiments that can satisfy the architecture, and the choice usually comes down to non-functional criteria such as performance, scalability, reliability, security etc.

The first-time reader might prefer to skip to the next section, which provides an overview of a working embodiment, and then cross-reference the embodiment with the architecture – that is, there is a gap between architecture and embodiment which is not trivial to bridge.

7.1 The Transport Session

A *transport session* represents the communication mechanism used to exchange data between session entities. Data can be both real-time, and non-real-time. A transport session could be implemented by an IP multicast group, a unicast IP conference, or, as in the current embodiment, a tightly-coupled, group communication server. A transport session is created by the communication session to establish the communication transport between session entities.

By *transport session state* we mean the collective attributes of a transport session: the connection, authentication and authorization parameters required to join in the data transport

mechanism, the set of channels in the transport session, and the set of channel endpoints connected to the channels.

A *transport session factory* is an abstraction of the notion of creating a transport session of an appropriate type. Requests to the transport session factory to create a transport session are parameterised with the characteristics of the required session. The transport session factory uses this information to create an instance of a specific transport session that satisfies those characteristics.

7.2 Media Channels

A transport session is a container for one or more *Channels*. A *Channel* is an instance of a multi-party communications path between *Channel Endpoints*. An instance of a channel is created within the context of a single transport session. A channel has a unique name within the transport session. A channel defines a communication path between the connected channel endpoints, that is orthogonal to all other channels within the transport session.

A *Channel Endpoint* is an instance of an addressable communication source or destination. A channel endpoint has a unique name within the context of a channel. An instance of a channel endpoint is bound to a single, named channel. An application communicates by first creating an instance of a channel endpoint and binding it to a channel. The application can then use the channel Endpoint to send data to and receive data from other channel endpoints connected to the Channel. Media applications communicate by exchanging data along the communication paths defined by Channels.

There are three modes for sending data on a channel from a channel endpoint:

1. Data is sent to all channel endpoints connected to the channel, including the sender.
2. Data is sent to all channel endpoints connected to the channel, excluding the sender.
3. Data is sent to a specific channel endpoint in the channel, by specifying the channel endpoint address. No other channel endpoints connected to the channel receive that data.

Typically, a channel is used to disseminate information of a given *media type*. Examples of media types are textual chat, voice chat, shared whiteboard, collaborative browsing, and real-time voice and video.

A *media client* is an instance of a specialized function to send and receive data of a specific media type, using a channel endpoint bound to the channel for the media type. The *media client* offers a specialized programmatic interface to the channel for communicating with other media clients of the same type. When a media client is created, it will create a named channel appropriate for the specific media type, if one does not already exist. The media client then tries to join the channel by creating a channel endpoint and binding it to the channel.

The *media description* specifies the number and media type of channel instances in a transport session, and the connection details required to join the transport session. The transport session connection details include the address and type of the transport session, and the authentication and authorization information required to join the transport session. The Session Description Protocol (SDP, RFC 2327) defined by the IETF is a good example of a standard mechanism to specify the media description. The current embodiment uses a much simpler scheme for the media description, that identifies a set of media clients to be instantiated by media applications, the address of the transport session, and the authentication information required to join the transport session.

An application may join any number of sessions. When an application joins a session, it uses the media description to decide the set of channels to bind to. Typically, the application creates one media client for every media type supported by the session.

7.3 The Leg Controller and Connection State

The *connection state* of a session entity is a representation of the state of the connectivity of the session entity to the communication session. A small number of states are defined that model the stages of joining and leaving a communication session. The states correspond to the notions of inviting a called entity to a session, alerting the called entity of the invitation, connecting to the transport session, requesting a disconnect from the transport session, and being disconnected from the transport session.

A *leg controller* models the signalling, events, and state machine used to invite a remote entity into a transport session and subsequently change and monitor the connection state of the entity. The procedure of inviting a remote entity into a session involves a signalling exchange between a pair of leg controllers. One leg controller is used by the inviter, the other by the invitee. Both leg controller instances only exist for the duration of the participation of the invitee in the transport session.

The leg controller supports a small number of operations, and generates a small number of events, used both by an inviter and invitee. The leg controller operations are used to communicate to the peer the progress of joining or leaving the transport session. The leg controller events represent the asynchronous notification received from the peer leg controller.

The inviter creates an instance of leg controller and uses it to invite the remote invitee into the transport session. The description of the transport session is passed in the invite operation. The invite request is directed to a *connection endpoint* located at the client of the invitee. Every connection endpoint has a unique address used to identify it. The address is used in a similar way as a telephone number in the PSTN. On receiving an invitation request, the connection endpoint instantiates the invitee leg controller.

The signalling exchanged between leg controllers carries a variety of data, the most important of which is the media description of the transport session. The abstractions presented by the leg controller represent high-level, logical participation in the transport session, and are independent of the communication mechanism used by the transport session. When an entity connects to, or disconnects from, a transport session, the leg controller is used to signal to the peer leg controller.

The external interface to, and logical signalling used by, the leg controller is independent of the mechanism used to transport the signalling information. Many different transport mechanisms for leg controller messages are possible. Two transports have been implemented in the current embodiment. One uses Java JMS, the other is a special case to enable communication to the customer desktop through a firewall that uses the communication primitives of the Transport Session. Socket and SIP transports are good examples of possible alternative implementation choices.

7.4 The Communication Session

A communication session is a representation of the *state* of a *set* of *communicating entities*.

An *entity* will in most cases be a person, although our architecture permits several kinds of software automata to participate. These are referred to as *Bots*, and are described in more detail below. A person will communicate via some kind of proxy interface, such as software running in a WWW browser.

By *communicating* we mean that entities are using one or media types to communicate, such as speech (audio), video, plain text, diagrams and illustrations, graphics, animations and 3D content, the kind of communications that are appropriate between human beings who want to share information.

By *set* we mean the normal mathematical notion of a set, a collection with a set of operations for adding a member, removing a member, enumerating members etc. These operations are specified in more detail below.

By *state* we mean the collective attributes of a specific session: the identities of the communicating entities, the media types in use, the pattern of distribution of information, global session properties (such as admission criteria), privileged members of the set etc. Specific examples of these session attributes are given below.

A *communication session factory* is an abstraction of the notion of creating an empty communication session, and of destroying a communication session.

The communication session has several functions:

1. Maintain the set of session entities. The communication session maintains the logical set of session entities currently in the session, together with their connection state to the session. Session entities can added to or removed from the communication session, using a small number of operations made on session entities.

2. Create a transport session. The communication session uses the transport session factory to create a transport session. The transport session is created in a lazy fashion, only when required. The reference to the transport session forms part of the state of the communication session.

3. Operations on session entities. The communication session has a small number of *session operations* that can be performed on session entities. The basic operations add and remove session entities to the communication session, and transfer session participation from one session entity to another. The communication session implements the session operations by translating operations into a sequence of operations on instances of leg controller to change the connection state of the affected session entities. The operations can be described in terms of a simple leg algebra of addition and subtraction of legs into the communication session.

The communication session operations are used to explicitly add or remove session entities. Session entities can also perform simple autonomous operations that affect the communication session. A session entity can autonomously *Join* a communication session, if the connection details of the communication and transport sessions are known and the session entity has the appropriate privileges. Session entities can autonomously *Leave* the session by invoking the disconnect procedure of the leg controller. Both autonomous operations will update the state of the communication session, including the set of session entities, and the appropriate communication session events will be generated (*EntityAdded*, *EntityConnected*, and *EntityRemoved*).

7.5 Layering of Abstractions

Figure is a representation of the abstractions described. The scenario shown in the figure is of two session entities connected to the same communication session. Each session entity could represent a Bot or a communication desktop used by a human participant in the session.

The figure shows how the abstractions fit into conceptual layers of related functionality. As is typical for a layered model, the abstractions of the higher layers are implemented using the abstractions of lower layers. The four layers represent different logical views of the connectivity and communication between the various entities, each with different concerns and objectives.

The figure also shows a representation of how the elements from the various layers map to the functional components of the Web Collaboration Service.

The four layers of the model are described below:

The *Service Layer* represents the service logic written by application developers to intelligently conference session entities into a communication session. A service manipulates the connection state of a set of session entities to a communication session, using only the abstractions of the Communication Session Layer. A service uses the communication session operations to invite entities to or disconnect them from the session, and uses the communication session events to monitor changes in connection state of the session entities.

The *Communication Session Layer* offers a high-level view of the participation of session entities in a communication session. Users of this layer deal only with very high-level abstractions of participation in a conference. The communication session, communication session factory, and session entity are the principal abstractions offered by this layer. The communication session and session entity uses the leg controller to invite the remote participant to the transport session.

The *Connection Layer* represents the protocol, messages, events, state machine and operations used to invite a participant to a transport session, and subsequently manipulate the connection state of the participant to the transport session. The leg controller is the principal abstraction in the connection layer. The connect and disconnect procedures offered by the connection layer are independent of the implementation mechanisms used for the transport layer. The connection layer abstractions use the operations, and consume the events offered in the transport layer. The events generated by the connection layer are used by the communication session layer to update the state of session entities and the communication session.

The *Transport Layer* represents the set of abstractions for the transport mechanism to exchange application data between session participants. The transport session is instantiated by the communication session using the transport session factory. Channels and channel endpoints can be instantiated by any entity with sufficient privilege. In the current implementation, channels are instantiated by entities in the connection layer, and by media clients.

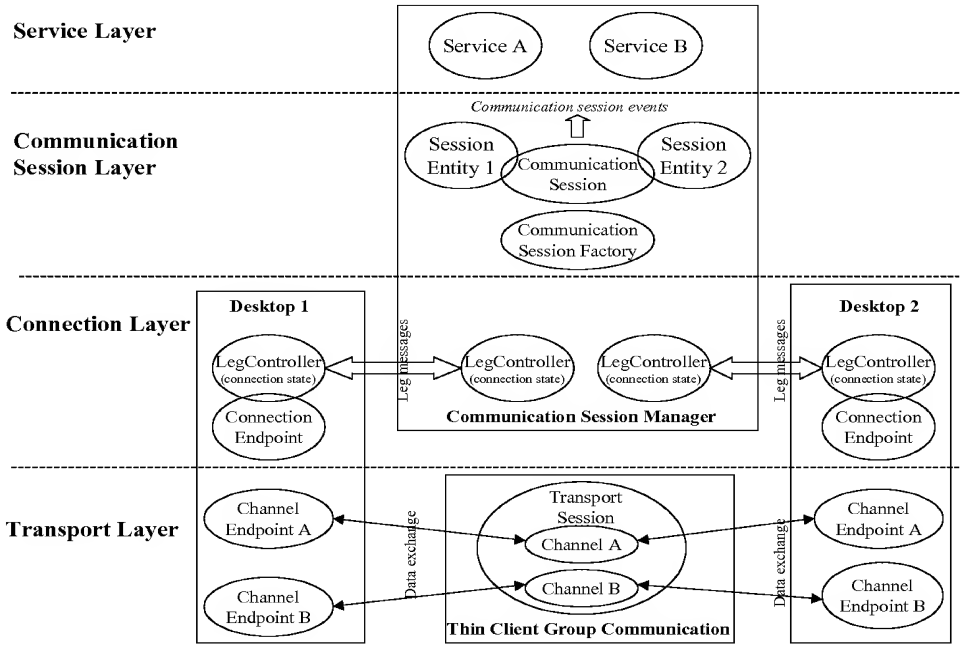


Figure 17: Layered Abstractions

The messages exchanged between functional entities in the connection layer can contain information from other layers, in addition to the specific information of the connection layer

itself. Services in the service layer are able to pass arbitrary information as key-value pairs to session participants in the add and remove operations. The communication session layer uses the transport layer messages to send information to invited session participants describing the current state of the communication session abstractions, allowing session participants to reconstruct the current view of the communication session layer.

7.6 The Service Interface to the Communication Session Layer

The services in the service layer only deal with the abstractions of the communication session layer, principally the communication session factory, the communication session, and the session entity. The operations supported by and events generated by the communication session layer abstractions are described in this section.

7.6.1 Communication Session Set Operations.

The operations that can be performed on the communication session factory are described below:

CreateSession(MediaDescription)	Create a communication session capable of hosting the specified media description.
EnumerateSessions(Pattern)	Enumerate all currently active communication sessions that match the characteristics of the specified pattern.
Destroy()	Disconnect all participants, and destroy the communication session and the associated transport session.

The basic operations that can be performed on the communication session are described below.

Invite(SessionEntity, InviteParameters)	Invite the specified session entity to the session, passing the specified invite parameters in the invitation message, and add the session entity to the set associated with the communication session. The invite parameters are name-value pairs that can carry arbitrary information to be passed to applications of the called participant. The transport session is created if not already in existence. The session entity is invited in to the transport session using the *invite* leg controller operation.
Remove(SessionEntity,DisconnectParameters)	Disconnect the specified session entity from the session, passing the disconnect parameters in the disconnect message. The disconnect parameters are name-value pairs that can carry arbitrary information to be passed to applications of the disconnected participant. When the disconnect procedure has completed, remove the session entity

	from the set associated with the communication session.
Transfer(TransfereeSessionEntity, TransferedSessionEntity, InviteParameters)	This is a compound operation. Transfer a connection to the session from the transferee session entity to the transferred session entity. The operation is completed successfully when the transferred to session entity has connected to the session, and the transferee session entity has disconnected from the session.
GetSessionAttributes	Get communication session attributes, such as the reference to the transport session and media description of the communication session.
EnumerateSessionEntities	Enumerate the set of session entities in the session. The connection state of each of the session entities in the session can be queried.

7.6.2 Communication session events

An application can register with the communication session to receive *communication session events*. The communication session events generated by the communication session are described below:

EntityAdded(SessionEntity)	The specified session entity has been added to the session, but is not yet connected. The set of session entities in the session is updated.
EntityConnected(SessionEntity)	The specified session entity has connected to the transport session. The connection state of the specified session entity is updated.
EntityDeclined(SessionEntity, DeclinedInformation)	The specified session entity declined invitation to the communication session. The declined information gives additional information, for example the reason why the invitation was declined.
EntityHeld(SessionEntity)	The specified session entity has placed the communication connection on hold, and is therefore not exchanging data with other members of the session.
EntityRetrieved(SessionEntity)	The specified session entity has taken the communication connection off hold, and is therefore again exchanging data with other members of the session.

EntityRemoved(SessionEntity)	The specified session entity has been removed from the communication session. The connection to the transport session has also been removed and the connection state and leg state updated appropriately.
AllEntitiesRemoved	All session entities have left the session
EntityTransferred(TransfereeSessionEntity, TransferedSessionEntity, TranferInfo)	A transfer operation has completed between from the transferee session entity to the transferred session entity. TranferInfo contains information about the outcome of the operation, such as a success result, or reason for failure.
SessionDestroyed	The communication session has been destroyed and all session entities have left the session

7.7 Session Routing

The architecture of the Web Collaboration Service allows considerable flexibility in how a request from a user to communicate with one or more other participants is satisfied. The request is addressed to an abstract *routing entity*. The routing entity may represent a room, a service, or a person. In fact the routing entity may represent any concept that is meaningful both to the requesting party and the routing services that handle the requests.

In general, the establishment of communication between session entities is a two-step routing process. The first step is to select a communication session, either pre-existing or created for the new call, for the calling party to join. The second, optional, step is to extend the participants in the communication session by selecting another party to invite to the session. The routing entity can be used in both routing operations to select the appropriate communication session and session entities. The routing entity maps, via arbitrarily complex logic executed on behalf of the calling party, to a communication session that hosts the communication with other parties associated with the same routing entity.

The routing logic for both steps can be arbitrarily complex, and use a configurable set of rules. The routing logic is parameterized with an arbitrarily large amount of information about the calling party, groped into related sets, such as the routing entity, and the requested service. The related sets of parameterized information are described below:

1. Calling party. This data set is used to describe the characteristics of the calling party. Examples are the name, e-mail address, physical address, country of origin, telephone number, gender, and profession. Other attributes could relate to preferences of the calling party, such as an interest in sport and music. An important attribute of the calling party is a unique user identifier, used by the system to identify the calling party, and used as a key to a persistent database of personal information held about the customer. Typically, this unique identifier is created the first time the calling party visits the site, and identifies the calling party for all subsequent interactions with the site.

2. Communication endpoint. This data set is used to describe the communicating device used by the calling party, for example the media capabilities and name of the device.

3. Called routing entity. This data set is used to describe the entity the calling party wishes to establish a communication session with. For example, the abstract entity may simply be 'customer service representative', with additional attributes that describe a marketing campaign such as 'Vacuum cleaners'.

4. Service data. This data set is used to describe additional information required by the specific service associated with the selected called abstract entity. A request to communicate with an abstract entity is handled by software servers, associated with the Web site, running services activated by requests from calling parties. For the example of the 'customer service representative' abstract entity, described by the 'Vacuum cleaners' campaign, the service data could correspond to a specific product range, feature set, or price range.

5. Communication option. This data set describes to the preferred communication mechanism of the calling party. The calling party may wish to communicate by Internet or non-Internet channel, or some combination of the two. Non-Internet channels could be telephone, or fax. Internet channels represent a variety of multimedia data types such as text or voice chat, collaborative web browsing, Internet voice and video telephony. The communication session may be established either immediately or at a future time, specified by the calling party. Often the communication option is service-specific. For example, the calling party may want to communicate with other individuals with similar interests, with a customer service representative, or with an automated bot.

The sets of parameterized data, described above, are derived and collated from several sources:

1. Embedded in a Web page. The Web page returned to calling party and used to request the establishment of a communication session, may contain arbitrary amounts of embedded data in the form of name-value pairs. The data may be statically embedded in the page, or dynamically generated by active server technology, such as Java Server Pages (JSP), as understood by those skilled in the art. This data is extracted and passed in the request for a communication session to the Web server. The data may be visible or invisible to the calling party, the decision is made by the web site designer. For example, the routing entity described by 'customer service representative' and 'Vacuum cleaners' campaign may be invisibly specified in the web page.

2. Input from the called party. The called party may be presented with a form to input information about him/herself, to describe the communication option required, or to select the abstract routing entity.

3. Persistent data in calling party browser. Information can be stored in the customer browser, to describe or simply identify the customer, to maintain service state, or calling party preferences.

4. Persistent information in a database server. Calling party information can be persistently stored a database server. The calling party identifier is used as the key to the appropriate database entries.

5. Dynamically generated service information. The service invoked when the calling party makes the communication establishment request may use any of the data described above to dynamically generate additional data to be passed to the session routing function to make the session routing decision.

Service logic executed on behalf of the calling party selects a pre-existing communication session, or creates a new one if no appropriate session is available. A description of the communication and media description of the transport session is returned to the calling party, such that communication applications operating on behalf of the calling party can connect to the session and communicate with other parties in the session.

7.8 Functional Entities of the Web Collaboration Service

The architectural entities described in this section are grouped into a number of functional entities. The characteristics of each of the main functional entities of the current embodiment is described in more detail below.

7.8.1 Thin Client Group Communication Server

The Thin Client Group Communication (TCGC) server hosts a set of transport sessions. One or more TCGC servers can be instantiated, to provide scalability. The TCGC server exports a simple interface to the transport session factory to allow other functional entities to create and destroy new transport sessions. The transport session is created with a set of authorisation parameters which are passed across the factory interface. In the current embodiment, the Communication Session Manager is the only client of the transport session factory interface.

The transport sessions offered by the TCGC server are centrally managed. The TCGC server is responsible for authenticating entities that attempt to perform operations on the transport session, channels and channel endpoints. In particular, transport session creation and destruction, creating or obtaining a reference to a channel, and binding a channel endpoint to a channel, are all operations that need to be authenticated by the TCGC server. The TCGC server has full knowledge of the set of channels created in a session, and the set of communication endpoints bound to the channels.

Communication between channel endpoints is implemented using a unicast transport. The communication endpoint sends data to the TCGC server, and it is the server that forwards the data on the unicast transport to each of the destination communication endpoints.

7.8.2 Communication Session Manager

The Communication Session Manager (CSM) provides a platform on which to deploy services to create and manage communication sessions, and manipulate the set of participants within a session. The realisation of the communication session, communication session factory, session entity, and server-side leg controller abstractions are part of the CSM. The CSM holds the definitive view of the communication session state.

The interface used by services is a Java API to the communication session abstractions, session entity, and communication session factory abstractions. Consequently, a service is shielded from the details of layers below the communication session layer. More details of the service interface are given in section 7.6.

In the current embodiment, the CSM offers a Java RMI remote interface to allow the servlets running on the SMS to create and communicate with instances of the set of services deployed on the CSM.

7.8.3 Session Mediation Server

The Session Mediation Server (SMS) hosts the set of services that present the GUI to allow a calling party to request connection to a communication session. The GUI allows the calling party to select a service, enter personal information, select a communication option, and

describe the entity to communicate with. The SMS uses the RMI interface to the services running on the CSM to satisfy the communication requests of the calling party.

In the current embodiment, the SMS services are implemented using Java servlets and JSP pages, as understood by anyone skilled in the art.

8 Glossary

ACD Applet Customer Desktop

CD Customer Desktop

CSM Communication Session Manager

CSR Customer Service Representative

DMZ De-Militarized Zone

HTTP HyperText Transfer Protocol

JMS Java Message Service

LCD Lite Customer Desktop

PSTN Public Switched Telephone Network

RMI Java Remote Method Invocation

SMS Session Mediation Server

TCGC Thin Client Group Communication

URL Uniform Resource Locator

WWW World Wide Web

9 INTELLECTUAL PROPERTY GENERATED BY PIRANHA PROJECT IN

Piranha Project Team, HP Labs Bristol

Date:

Version: 0.1

Document purpose: Provides a self-contained description of the architecture of the HP Labs Internet Interaction Server and its application to Customer Relationship Management in the SmartContact/Web Collaboration product.

The basic approach in the first draft of the document is to describe the system, its components and its characteristics as concisely as possible, and to describe the underlying intellectual property in enough detail to enable a discussion with IP attorneys about patentability.

9.1.1.1.1.1.1.1 Identifies the intellectual property generated, separated into

- ✓ IP that has been implemented in SmartContact/Web Collaboration
- ✚ IP which is independent of SmartContact/Web Collaboration.

9.1.2 Background ***Colin***

Section purpose: describes field of application. Shows how CRM is just one part of a bigger picture.

9.1.2.1.1.1.1.1.1 Architecture *Colin*

Section purpose: General description of system model and architecture. Identifies which components have been built in Web Collab. Identifies specific architectural notions which should be considered for patentability. Ideally also identifies a toolkit/set of reusable components which can be applied more generally than Web Collab to support multiparty multichannel internet interaction sessions.

9.1.2.1.1.1.1.1.2 Multiple Consecutive Communication Methods

10 Multiple Concurrent Communication Methods *Rych and Lawrence*

- ✓ Sessions and Channels.
- ✓ Media description.
- ✚ Integrating other media types – SIP/STP, 3D

10.1.1.1.1.1.1.1.1 Multi-party Communication Sessions *Lawrence and Colin*

- ✓ Transport independent, Application independent.
- ✓ Multiparty session invite/join/transfer/conference
- ✓ Leg model and algebra
- ✚ Sessions involving bots (agents or robots automating a role or performing a function), real agents, and real people

10.1.1.1.1.1.1.1.2 Multiple Session Initiation Models *Colin*

- ✓ Basic contact methods: callback now, callback deferred, fast connect, internet rendezvous etc. (note: Internet rendezvous is a novel contact method).
- ✚ Shopping with friend(s), agent(s) and bot(s).
- ✚ Meeting an agent/friend on a Web page.
- ✚ Joining a nominated session.
- ✚ Unsolicited invitation to join a session.

10.1.1.1.1.1.1.1.3 Session Management (Modification and Routing) *Lawrence*

- ✚ Supervisors can join sessions in different modes.
- ✚ Routing to pre-existing session.
- ✚ Intelligent routing based on customer profile.

10.1.1.1.1.1.1.1.4 Security *Hans*

- ✓ Tunnelling persistent communication sessions (PATENT APPLICATION FILED)
  - Other IP in firewalls/proxy server configurations etc ?

10.1.1.1.1.1.1.1.5 System Components

Section purpose: Goes into more detail about system components architected (but not necessarily implemented) in Web Collaboration. Identifies specific work which should be considered for patentability.

10.1.1.1.1.1.1.1.6 Agent Desktop *Rych*

- ✓ Icons popping up when media appears on a channel
- ✓ Multi-call control
- ✚ Multimedia scripts – pushing to multiple channels
- ✚ 3D synchronisation

10.1.1.1.1.1.1.1.7 Customer Desktop *Colin and Rych*

- ✓ Lite desktop: text chat and page push with no modification to customer browser.

10.1.1.1.1.1.1.8 **Bots** Colin and Rych

- ✓ Marketing on hold *Colin*
- ✚ Content monitoring bot *Rych*
- ✚ Transcript bot *Colin*

10.1.1.1.1.1.1.9 Transcript Repository *Hans*

- ✚ Multichannel transcripts

10.1.1.1.1.1.1.10 Wake Repository *Hans*

- ✚ URL wake logging

Appendix 1 – Potential Patent Claims

1 Customer Help Session – Basic Case

1.1 Introduction

A customer is browsing a WWW site and wants to interact with a CSR. The WWW page contains some means (e.g. a "Help" button) to initiate a session initiation dialogue. The customer uses the WWW to submit a small amount of personal information, then initiates a session. Appropriate media interfaces are created. The customer joins the session, a CSR is allocated and also joins the session.

1.2 Detail

The initial customer dialogue collects the following information:

- Referror URL. This is the page from which the help request is made. This information can be extracted from the HTTP header. It is presented to the CSR to show the page that prompt the customer to request help.

- Cookies. A number of cookies can be set for that customer during a previous interaction. Typically a CustomerId cookie is sufficient, as that can be used to reference a customer database and so extract information about the customer and previous interactions.

- Campaign. Information about the campaign can be embedded in the WWW page. For example, a WWW page about freezers could have the campaign set to "Freezers". In our embodiment this information is encoded in the URL used to request Help as an HTTP query string. In this way the customer service request can be routed to a CSR in the "Freezers" campaign, who is presumed to have specialized knowledge about freezers.

- Locale. This information as located on the WWW page as a query string, and is used to select customer messages based on Locale and language information. For example, a customer in France could receive messages in French. This information could also be used to route to a French-speaking CSR.

- Nickname. This is the name by which the customer is known in the communication session.

- Personal Information. First name, last name, title, email address – these are examples of information that enable the CSR to be more courteous. It is up to the enterprise how much information of this kind is collected.

Personal information extracted by the Session Mediation Server (SMS) is communicated to the Communication Session Manager (CSM) which sets up an empty communication session and an empty transport session. The CSM replies with information about the session (including the session's secret name and password) and this is dynamically embedded in a WWW page which is returned to the customer by the SMS. In our embodiment Java Server Pages is used to embed information, a technique understood by those skilled in the art.

When the page is returned to the customer's browser it contains active content (either Java or Javascript) that is capable of creating media interfaces that can use session information embedded in the WWW page to join the transport session.

The CSM needs to find a CSR to help the customer. It does this by using an external library interface that enables it to request a free CSR from a pool of CSRs. This request can take some time (there may be no free CSRs) and the CSM receives regular events which indicate the position of the customer in the queue of waiting customers. This information can be forwarded to the customer using a control media channel, and displayed on the customer desktop, so that the customer is regularly informed of the current delay in allocating a CSR.

While the customer is waiting, a bot can join the session to provide the "Marketing on Hold" service, by pushing marketing or entertainment content at the customer.

When a CSR is allocated, the CSR desktop software receives an *invitation* to join the session. This causes a visible event on the CSR desktop. The CSR may accept or decline. If the CSR accepts, then the CSR desktop uses information about the session to join the session. At this point the CSR and customer are now in communication.

There are many possible error terminations. The customer may fail to join the session. An CSR may not be available. The CSR may decline the session. Both customer and CSR may leave the session prematurely without warning due to software or hardware failure. The CSM must monitor the session for changes in membership, so that resources can be cleaned up.

During a session the CSR can

- Transfer the "call" to another CSR.
- Request another CSR to be included in the session.

1.3 Prior Art

There is no novelty in the broad service. A number of vendors provide this kind of Help session.

1.4 Claims

The following ideas may form the basis for patent claims:

- Marketing on Hold (see Overview and above).
- Customer queue information sent to the customer (see above).
- Transfer and conference.

2 Customer Help Session – Telephone Callback

2.1 Introduction

A customer is browsing a WWW site and wants to interact with a CSR. The WWW page contains some means (e.g. a "Help" button) to initiate a session initiation dialogue. The customer uses the WWW to submit a small amount of personal information, including a telephone number, then initiates a session. Appropriate media interfaces are created. The customer joins the session, a CSR is allocated and also joins the session.

A PBX associated with the CSR is used to set up a third-party telephone call between the CSR and the customer.

2.2 Details

The details are as described in section 1.2. Once the CSR has joined the session, a third-party telephone call is initiated to the customer's telephone number, which is supplied as part of the initial dialogue. The call is initiated by the CSM using a library interface to a telephone switch, this is usually the enterprise PBX required as part of a call center.

2.3 Prior Art

As per section 1.3.

2.4 Claims

The following ideas may form the basis for patent claims:

- Marketing on Hold (see Overview and above).
- Customer queue information sent to the customer (see above).
- Transfer and conference.

Customer Help Session – Deferred Telephone Callback

2.5 Introduction

A customer is browsing a WWW site and wants to interact with a CSR. The WWW page contains some means (e.g. a "Help" button) to initiate a session initiation dialogue. In this case the customer wants to interact at some future time, and so asks for the session to be deferred. The customer uses the WWW to submit a small amount of personal information, including a telephone number. At the conclusion of the dialogue the customer is requested to go to a URL (www.thisorg.com/fast.html say) when the telephone callback occurs.

The CSM places the request in a list of pending requests and scans it periodically. When the callback time occurs, it makes a telephone call to the customer. The customer picks up the phone, and then has the choice of supplementing the telephone call with other media using the WWW as in section 0. The customer can then use a WWW browser to go to the previously mentioned URL. This initiates a session as in section 1.2 and section 2.2 so that the customer has the full range of media interfaces.

2.6 Details

Service initiation is divided into two parts. The first part is as described in section 1.2, but instead of joining a session, the customer is directed to go to a URL when the telephone callback occurs. Unknown to the customer, a "cookie" is sent to the browser as part of the HTTP response header during the first phase of initiating a session. The cookie contains a session identifier for the customer.

When the telephone callback occurs and the customer goes to the previously specified URL, the cookie is extracted from the HTTP header by the SMS and forwarded to the CSM. The CSM uses the session identifier in the cookie to identify the customer, the pending request, and the previously acquired customer information. This information is returned to the SMS, which is then able to construct an HTML page containing the information about the session as in 1.2, and the customer browser is able to create media interfaces for the session and so join the session.

There is no need for the CSM to find a free CSR, as the CSR is already allocated to the call. The CSR receives an immediate invitation to join the session, and joins as in 1.2.

2.7 Prior Art

I am not aware of prior art. This service may be novel.

2.8 Claims

The customer interaction is divided into two parts, the first part being used to acquire information, the second being used to join a session. A session identifier is used to link the two phases. This identifier could be given to the customer explicitly, but we use the cookie mechanism to automate the presentation of the session identifier.

3 Customer Help Session – Telephone Callback Only

3.1 Introduction

As for section 2.5, with the exception that the customer desires a telephone callback only. No URL is supplied.

3.2 Details

As for section 2.6. The customer receives a telephone callback, but no WWW interaction session is required or started. In our embodiment the implementation is identical to that in section 0.

3.3 Prior Art

There is no novelty in this service. There are many existing "click for callback" services active on the WWW.

4 Customer Help Session – Rendezvous

4.1 Introduction

The customer is talking to a CSR on the telephone, either as a result of direct dial or callback. No prior history of interaction is assumed. The CSR suggests that the customer can use the WWW to improve communication, and gives the customer a session name (handle) and password. The customer goes to a WWW page, enters the session name and password, and joins a newly created session. The CSR receives an invitation to join the session.

4.2 Details

The implementation details for this service have already been described in section 1.2. The difference is that in 1.2 a CSR has to be allocated, while in this service a CSR is already allocated. In our embodiment the session name (handle) is identical to the CSR name, which make locating the correct CSR trivial.

The session password is generated dynamically by the CSM as a one-time ticket. It is generated by the CSR using an appropriate desktop GUI, which calls the CSM using a request-response protocol such as Java RMI. It takes the form of a short string of digits which can be read out to the customer over the telephone.

4.3 Prior Art

No prior art is known. This service and its implementation may be original. The idea came from requirements collected by CRMO from a beta-test customer.

4.4 Claims

The following ideas may form the basis for patent claims:

- A session is described using information that can be dictated over the telephone by the CSR. This unique information is used as a basis for rendezvous by customer and CSR. The normal CSR selection procedure is bypassed.

- A telephone session is extended to include WWW media channels.

5 Shop with Friends

5.1 Introduction

A group of customers want to interact while browsing the WWW. One customer, the session initiator, goes through a dialogue to set up a session, and is supplied with a name and password for the session. The session name and password are communicated by some means (e.g. email) to other potential session participants.

Each potential participant goes to a WWW page that has a form to input the session name and password. This results in the person joining the session.

Members of the session use follow-me page push to view the same WWW pages, and use media channels to interact. It is possible to request a CSR to join the session.

5.2 Details

A communication and transport session are created by the CSM in response to the request by the first customer. Each subsequent participant request is routed to the same session (see Session Routing in the Overview) using the (secret) session name.

A CSR joins the session by invitation by means already described in section 1.2.

5.3 Prior Art

The WWW site www.landsend.com has a two-participant "Shop with a Friend" service using a system supplied by Webline (www.webline.com). Due to technical limitations the session does not appear to be extensible to more than two participants. It is not possible to invite a CSR.

5.4 . Claims

The following ideas may form the basis for patent claims:

- use of our multiparty session model for communal browsing.
- Inviting a CSR into a session.
- Routing based on page, session or customer attributes to select an CSR.

6 Page as Place

6.1 Introduction

A customer is browsing a WWW site. On each page the customer can tell whether other people are looking at the same page, and can choose to engage them in conversation. Each WWW page can be thought of as a place shared with others. CSRs can observe the activity on a WWW page and can also make their presence known. In this service, each WWW page has the properties of a meeting room. This property can be extended to sets of pages.

6.2 Details

It is assumed that each customer has a set of desktop media channel GUIs (as described in the Overview) which persist as the customer moves from page to page. When the customer moves to a new WWW page, an element on the page (e.g. a graphic element) is loaded by a hyperlink that invokes a Java servlet on the SMS. This embodiment can be generalized to the many ways of running programs on HTTP servers. The Java servlet extracts the referrer URL from the HTTP request header, which identifies the WWW page, and notifies the CSM that a new person has entered that page.

The CSM replies with session information about the page, and the SMS causes the customer's media GUIs to join the session. The exact details vary for the Applet and Lite desktop embodiments.

When the customer leaves the page, a Javascript "onUnload" event can be used to trigger exit from the session.

Empty sessions associated with pages can be pre-created by the CMS (using a script), or can be created dynamically when a customer goes to a page. The latter strategy would be appropriate for sites with very large numbers of pages.

CSRs would be equipped with a Session Browser (implemented using dynamically generated WWW pages) to view all current sessions. A CSR could pick a page of interest and join that session. In our embodiment this implemented by using the SMS to provide views of active sessions.

An additional mechanism would enable agents to set triggers on pages, so that they automatically joined a session when a customer was looking at a page (i.e. when a session with membership >= 1 was active on the page). This would be useful for pages where proactive intervention was useful.

6.3 Prior Art

The CoBrow (www.cobrow.com) system uses the idea of detecting WWW page access to trigger a session associated with the page. They mention several applications, including virtual shopping, with sales assistants observing customers entering a virtual store.

6.4 Claims

The following ideas may form the basis for patent claims:

- letting a CSR specify triggers on pages.

- letting the CSR join session in a "stealth" mode where they could monitor conversations and intervene if the customers have a problem. (This is little more that what a shop assistant already does).

- associating various descriptive labels with a WWW page (e.g. Kitchenware, White Goods, Freezers) and using these labels in the call center router to select an available agent. These labels could be embedded in a page as query fields and extracted by the SMS when it detects someone on a page. A CSR would then be alerted based on their skills.

- customers can a select a proper subset of those on a page to form a new session, operating now in "Shop with Friends" mode.

The following ideas may be worth claiming for tactical reasons but may not be novel:

- Providing the means for a CSR to view and join ongoing WWW page.

- using a Bot (virtual representative) in place of a CSR to provide place-specific assistance, silently monitoring for particular keywords (e.g. "help me").

7 Session Routing

7.1 Introduction

In virtual communities, there may be many long-lived communication sessions, each with many participants. The participants of the communication session may join and leave the session independently of the life of the session itself. Each session within the virtual community will be used for a different purpose. A session can host multi-party discussions on specific topics, and grow simply by additional participants joining the session. In addition, a session can host intentional communication between two specific participants, similar to one telephone user calling the number of another user. A session may be private to a specific small group of users, or public to anyone wishing to participate.

Sessions may have intelligent behaviors associated with them such that the set of participants within the session is controlled and managed based on a number of configurable criteria. For example, a session to host a discussion between a group of customers about which model of car to buy can be expanded to include a suitable customer service representative if the customers need assistance.

The address used in a communication request from a user can refer to any abstract concept, called the routing entity, meaningful both to the calling party and to the routing services that handle the request. For example, the routing entity may represent a service, product, discussion group, person, or a specific communication session. Routing services typically map the routing entity to a session or to other participants. The session or participants may be either explicitly named by a user, or automatically chosen on their behalf using a number of selection criteria.

With perhaps hundreds or thousands of communication sessions running simultaneously, the choice of how to deal with a communication request from a calling party becomes very complex. The calling party may be routed to one of the existing sessions, or to a session created solely to satisfy the request. It may be appropriate to invite additional participants to the session in order to satisfy the request, or simply allow the calling party to join a session to communicate with others already in the session or wait for others to join.

7.2 Claims

The claim relates to recognizing that a request from a calling party to establish a communication session with other communicating parties is explicitly a two-step process. The first step is the selection of an appropriate communication session. The second step is the selection of additional communicating parties to invite to the communication session. The second step is optional and independent of the first.

Both steps need automated intelligent routing decisions to be made on behalf of a calling party: to select an appropriate communication session to join, and possibly extend the session with a selected set of additional participants. Traditional session architectures emphasize routing to another participant. In virtual communities, routing to a session is just as important.

The routing decisions are parameterized using a variety of different information, accumulated from a number of sources. It is expected that a web server, the Session Mediation Server (SMS) described in the architecture document, will typically act as a portal and entry point to a virtual community. The information used in the routing decisions is derived using web technologies, described below.

1. Referrer page parameters. Web pages held on an Enterprise HTTP server can contain URI links to entry points to the SMS, in order to let users request the establishment of a communication session. These pages can be authored to contain embedded data, sent as parameters in an HTTP request to the SMS. For example, the pages of an e-commerce site may contain links to the SMS to allow customers to join discussion groups, or contact an agent, about a product. The page may contain parameters that describe high-level semantics of the page, such as the 'Vacuum cleaners' campaign and a specific model number.

2. Referrer URI. The URI of the page that held the link to the SMS is sent in the HTTP request to the SMS. This URI represents a simplification of the browsing history of the calling party.

3. Full browsing history. The Enterprise HTTP server may keep semi-persistent history of the set of Web pages accessed by a user in the current browsing session. This history is called the Wake Repository. A reference to Wake Repository can be passed in the request to the SMS.

4. Calling party identifier and client state. The SMS uses cookies to persistently store, in the Web browser, a globally-unique identifier for the calling party, such that the calling party can be identified across separate interactions with the SMS. The cookie mechanism can be used to store any other information that needs to be held persistently across each of the steps of a communication request, or across separate communication requests.

5. Customer input. The SMS presents the calling party with a user interface to enter information and make service selections. Calling party details could be name, e-mail address, postal address, country, telephone number, age, gender, profession, and interests. The calling party is able to select the characteristics of the requested communication mechanisms, such as text chat, voice chat, page push, shared whiteboard, Internet voice, Internet video, and PSTN telephone call. Depending on the selected communication mechanisms, the calling party may be able to defer the communication to a specified later time. Most importantly, the calling party can specify the characteristics of the abstract routing entity with which to establish communications.

6. Calling party database. The calling party identifier is used as a key to a persistent database of information held about the calling party, such as name, address, country, telephone number, or service subscription options.

8 Multiple Session Management

Office workers deal with multiple communication sessions using different media every day, e.g. phone, fax, email, text chat, pagers, SMS, collaborative web browsing, etc. Often some of these communication sessions occur simultaneously. Each media has it's own interface and a number of functions that can be performed, some of them common to all media, some unique. If the user was presented with the same interface then handling communications via these different media should become easier or perhaps allow more sessions to be managed simultaneously.

A Multiple Session Management (MSM) interface would handle the functions common to these media, e.g. joining and leaving a communications session. Individual media channel interfaces would provide access to the functions specific to that media type, act as displays for its content and provide a terminal for input.

To achieve this the MSM interface should offer the following features:

- Display of all information associated with session to help user distinguish sessions, e.g. name of person(s) in session, name of session, etc.
- Clear notification when an invitation to join a new session is received.
- Ability to switch between multiple sessions with minimal fuss.
- Notification when content of a session other than the current one has changed.
- Reuse of display space for each session to minimise the amount of information the user must assimilate at one time.

In addition, the media channel interfaces need to support the following:

- Indication whether the channel is being used in the selected session.
- Provide a record of content exchanged on the channel for every session (the point at which the record is deleted is implementation dependent).
- Allow dynamic input and output configuration for a specific country and language (enables multi-lingual users to interact in native dialects).

In certain usage scenarios, e.g. customer support, it is often necessary to ask the same series of questions, show the same sort of content, etc., in many different sessions. To make this process more efficient, scripts can be used to coordinate the transmission of content on multiple media channels simultaneously. These scripts have the following characteristics:

- They can be entered at any point and played for any length of time, e.g. just a single event up to the whole script.
- Scripts are interpreted as user input and as such can be interleaved with user generated content during playback or after they have completed.

8.1 Embodiment

The functionality of the MSM is represented by a GUI window that contains a table with each row representing a session that the user is invited to, is currently involved in or has recently left. The row displays contact information associated with the session. New invitations are highlighted in red, the currently selected session is highlighted in blue. Associated with this main window are a number of other GUI windows, one per media channel client. As different sessions are selected, so the media windows show the content of the channel specific to that call (if used).

When content arrives on a media channel for a call other than the one currently selected, an icon representing that media type appears in the table row associated with the session. When the user selects a session, the media icons are cleared.

When the user leaves the session, the session's details remain in the in the table and when selected, the transcripts for each of the media channels are available from the respective GUI windows. When the session is removed from the table, the transcripts are deleted.

A country and language is associated with each session to indicate what language and formatting, e.g. date, to use in the session. When a session is selected, the media windows are configured to use the appropriate input and output conventions.

Scripts can be triggered from any media window by selecting the appropriate entry from cascading pull-down menus. When a session is selected, the script in the appropriate language is made available to the user. If a script for a given language is not available, then the default language, e.g. American English, is used.

8.2 Prior Art

Synchronized Multimedia Integration Language (SMIL - http://www.w3.org/TR/REC-smil): facility to author multimedia scripts. Standard players output all media content on same display, not individual displays as described above.

8.3 Claims

Worthy claims:

- One-click management of multiple sessions.
- Providing some visible and/or audible notification of when content on an unselected session changes.
- Dynamic configuration of GUIs and content, e.g. scripts, depending on session language.

Questionable claims:

- User controlled (initiation and duration) multimedia script output to individual media displays.

9 Content Monitoring Bot

In telephony-based call centers, supervisors monitor calls in order to ensure a certain level of service, ensure that neither CSR or customer is on the receiving end of any abuse, and deter the CSR from unnecessarily extending the length of the call, among other reasons. When a call center uses multiple media, this job becomes harder.

A content monitoring Bot can be used to help the supervisor in their job. This Bot performs a number of functions:

- Detect the presence of profanity or abusive terms and phrases.
- Detect the transmission/sharing of undesirable material or references to such material.
- Collecting statistics of the media channel, e.g. measure the response rate of the agent to new content.
- Analyze content for frequently asked questions and their responses.

For the first two functions, the Bot identifies whether the CSR or the customer was responsible for the indiscretion. The Bot then notifies the supervisor's desktop, providing a copy of the transcript so far and the suspicious content. The supervisor would then decide whether further action is necessary, e.g. joining the session, terminating it, etc., or to ignore the report. The Bot may also communicate with the Transcript Bot to ensure that this transcript is saved for further examination. Indeed the functions of the Content Monitoring Bot and the Transcript Bot may be merged into one process.

Statistics collection is a useful tool for measuring CSR performance. A long call duration with a low response rate might indicate an overloaded agent (two many simultaneous calls perhaps) and a long call duration with a high response rate might indicate an overly "chatty" agent.

The analysis of content can be used to improve CSR scripts, contributing to a knowledge base or even providing material for a Bot that mimics a CSR.

10 Transfer and Conference

10.1 Introduction

The dominant model for incorporating CSRs into communication sessions is one of invitation: a piece of software is used to keep track of the number of active CSRs, the number currently engaged in dealing with customers, and the number who are idle. Of those are idle, some may have more appropriate skills and training that others for dealing with a particular customer. This software, which is external to the system being described, has a simple interface where it is possible to specify the skill set of an CSR, and it will pick a free CSR. At this point the agent is invited into a session.

Session control models are still in development, and have typically been targeted at groups of peers (e.g. university researchers) who publish details of a session (time, IP addresses and ports, media etc) and it is up to each individual whether to join the session.

Invitations are closer to what happens when the telephone rings. A telephone rings because there is a special invitation protocol designed to notify a callee that there is an incoming call. This protocol requires the telephone equipment to support the protocol. In the case of the kind of multiparty, multimedia sessions used in this invention, this is equivalent to each CSR PC running software capable of responding to session invitations.

While the research community is developing invitation protocols for this kind of application (Session Initiation Protocol), nothing was available off-the-shelf that we could have deployed in our invention, and so we invented protocols adequate for our requirements.

10.2 Details

Each CSR terminal equipment (a PC) runs a piece of software which in our embodiment is called a **LegInitiatior**. A CSM can address messages to this component for each CSR. The **LegInitiator** corresponds to the well-known port concept familiar to those skilled in the art.

When a CSM needs to invite a CSR into a session, it sends a *Delivered* message containing details of an invitation to the **LegInitiator**, which spawns a **LegController** object to handle the invitation. The CSR desktop has one **LegController** for each outstanding invitation or active session.

The LegController alerts the CSR (using a notification) that an invitation to join a session has arrived. The CSR can *Accept* or *Decline* the invitation. If the CSR accepts, the LegController exchanges messages with a peer **LegController** object in the CSM to transition into the *Established* state, at which point the CSR is a full member of the session, and media channels are connected to appropriate media channels in the underlying transport session. A state transition model for **LegControllers** is shown in Figure TBD.

For conference, a new CSR is added to the session. An existing CSR selects one of the CSRs to join, and uses a desktop component to signal to the CSM that a new CSR is to join. The CSM then goes through the invite procedure for the new CSM, which if accepted, creates a new Leg connecting the new CSR to the session.

For transfer, the initial steps are the same as conference. A new CSR is invited into a session. When the new CSR joins the session (the Leg is established) the requesting CSR is disconnected from the session.

10.3 Prior Art

This basic ideas are well known in the multimedia community. Their application to this kind of call center situation may be novel.

10.4 Claims

We should claim the application of our session, leg and invitation model to the contact center as novel.

Iryna Ridchenko

From: Colin Low [cal@hplb.hpl.hp.com]
Sent:
To: Squibbs, Robert Francis; piranha@hplb.hpl.hp.com
Subject: IP Documents


patent-claims-overview.doc (1 ...
patent-claims-detail.doc (130 ...

Robert,

Here is our latest attempt at IP claims. The overview has been expanded a little by LW to add more stuff on the architecture. A new document consisting of detailed claims has also been attached. I expect we can probably brainstorm new ideas and claims, along the lines of some of the things you suggested at our last meeting, but for the moment we have plenty to work with.

Cheers,
Colin

EXHIBIT C

A mechanism for updating a customer on the progress of their call in a call centre over a web channel.

Inventors: Three Stooges + Gary Crosby

Problem

Having requested help, a customer is left waiting for a CSR to appear to deal with their specific problems. Without any feedback on how long they may have to wait, the customer may get bored or disheartened and abandon the call. By displaying regular updates on the progress of their call in the call centre, they will be reassured that something is happening and more likely to wait/less irritable when the call does get through to a CSR.

These progress updates will initially be either a position in a queue of calls waiting for a free CSR with the appropriate skills/knowledge and/or an estimated time to when such a CSR will become available. Once a CSR has been selected, the updates may also contain the name and other personal information about that CSR, e.g. a photo, so as to prepare the customer for call establishment.

At any point after the call has been allocated to a CSR, that CSR may choose to transfer the call to a colleague, either by name or using the automatic CSR selection process as described above. In both cases, more progress updates may be sent to the customer's desktop. In the former case, the CSR's details may be displayed immediately; in the latter, the process as described above would be used.

Although progress information is quite common in a telephony channel, the authors are not aware of the same mechanism being applied to a web channel.

Method

(Note that the following text is different to that found in the old appendix.)

Personal information extracted by the Session Mediation Server (SMS) is communicated to the Communication Session Manager (CSM) that sets up an empty communication session and an empty transport session. The CSM replies with information about the session (including the session's secret name and password) and this is dynamically embedded in a WWW page that is returned to the customer by the SMS.

When the page is returned to the customer's browser it contains active content (Java, Javascript or some other scripting language) that is capable of creating media interfaces that can use session information embedded in the WWW page to join the transport session.

The CSM needs to find a CSR to help the customer. It does this by using an external library interface that enables it to request a free CSR from a pool of CSRs. This request can take some time (there may be no free CSRs) and the CSM receives regular events that indicate the position of the customer in the queue of waiting customers.

This information is forwarded to the customer desktop using the control media channel present in the multimedia session established for the call. Upon receipt of this information, the customer desktop displays it in the most appropriate manner.

When a CSR is allocated, the CSR desktop software receives an *invitation* to join the session. At the same time the customer desktop is informed that a CSR has been selected, their name and any other pertinent information. The customer desktop may choose to display the information immediately or wait until it knows that the CSR will actually take the call (see

below). Receiving an invitation causes a visible event on the CSR desktop. The CSR may accept or decline.

If the CSR accepts, then the CSR desktop uses information about the session to join the session. At this point the CSR and customer are now in communication and the progress update messages are no longer displayed on the customer desktop.

If the CSR declines, then the CSM uses the external library interface to request another CSR. At this point the customer desktop may display an appropriate apology message if necessary and then continue to display progress update messages starting with position in queue. Note: declining a call should be a rare event, since its use implies that the call was incorrectly classified in the first place. It is therefore more acceptable to display the CSR's details to the customer and risk that choice being invalidated than not to show it at all.

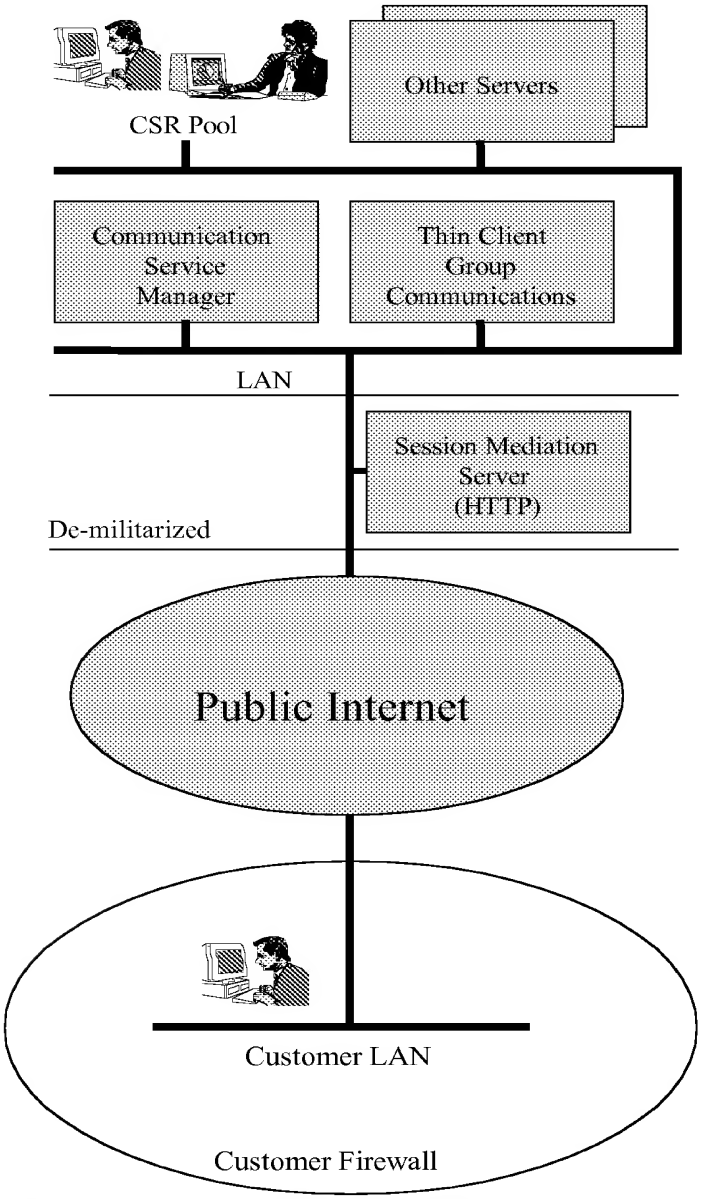


Figure 1. Customer Desktop Connection to Infrastructure

A method of delivering personalised advertising to the customer while they wait to either initiate or resume communication with a CSR, by generating (possible interactively) multimedia content on the most appropriate media channel(s).

Inventors: Three Musketeers + Gary Crosby (marketing from routing script) + Ram??

Problem

Having requested help, a customer is left waiting for a CSR to appear to deal with their specific problems. This may take some time and although they may be given feedback about how long it will take for a CSR to answer their call, more efficient use of that wait time could be made by presenting the customer with content of potential interest to them.

There may be a personal profile associated with the customer, constructed from previous interactions with the call centre and/or provided by the customer through use of a form. This profile can be used to personalise the content shown to them. In addition, if their movements throughout the web site have been tracked prior to requesting help, this information can also be used to select appropriate material.

This content could be for display only, in which case a periodic update will not only enable a range of information to be presented to the customer, but will also reassure the customer that their machine has not been disconnected from the call centre. Alternatively, the content may invite the customer to interact with it, e.g. conversing with a bot, reviewing a product presentation, etc., providing a more involving experience and reducing the likelihood that the customer will abandon the call through boredom.

Once the customer has been connected to a CSR, this content is removed from the customer desktop. At any point after the call has been allocated to a CSR, that CSR may choose to transfer the call to a colleague. This provides another opportunity to push content to the customer while they wait to be reconnected.

Method

Personal information extracted by the Session Mediation Server (SMS) is communicated to the Communication Session Manager (CSM) that sets up an empty communication session and an empty transport session. The CSM replies with information about the session (including the session's secret name and password) and this is dynamically embedded in a WWW page that is returned to the customer by the SMS.

When the page is returned to the customer's browser it contains active content (Java, Javascript or some other scripting language) that is capable of creating media interfaces that can use session information embedded in the WWW page to join the transport session.

The CSM needs to find a CSR to help the customer. It does this by using an external library interface that enables it to request a free CSR from a pool of CSRs. This request can take some time - there may be no free CSRs.

There are three basic ways of sourcing content for the customer during this phase:

1. Content is driven by the same component that is responsible for locating a free CSR.
2. A special Bot joins the communication session for either interactive or non-interactive content generation.

In both cases, the content is sent to the customer desktop using a media channel present in the multimedia session established for the call. Upon receipt of this information, the customer

desktop displays it in the most appropriate manner. The content may be a reference to the information to be displayed or the data itself.

The media channel that is used and the way in which the content is displayed on the customer desktop is completely open. If the content is a URL then it may be displayed in an existing window on the customer desktop, e.g. the page push window, or a new window might be opened to display the content (and any subsequent updates). If the content is being driven by a script or non-interactive Bot then a similar choice can be made. If the content is being produced by an interactive Bot then either the chat channel could be used and the existing chat user interface leveraged or a separate window opened for the new content which may also be distributed on a dedicated channel. Indeed, content may be transmitted on multiple channel simultaneously, e.g. a text description of the accompanying a 3D product visualisation.

In solution 1, the system component responsible for selecting a CSR already has a communication path to the customer desktop via the CSM and the media control channel. Solution 2 uses a communication client to join the appropriate channel so that the content can be sent to the desktop. The content may be generated by a third-party application, but the Bot understands the protocol used on the given channel and ensures that the content is sent in an appropriate manner. The Bot exploits the Communication and Transport Session models described in Section 3.2. The Communication and Transport Session layers are exposed by an application programming interface that enables the implementers of the CSR and customer desktop components to access the services provided by these layers, such as session state notifications (e.g. participants joining and leaving), participant information, and media delivery. The Marketing-on-Hold Bot is a software package that joins a session in lieu of a CSR using the same interfaces, and uses these interfaces to exchange media content with the customer, using the same media channels as would be used subsequently by a CSR. All the media clients (see section 6.5) in the customer desktop are exposed in this way. Anything a CSR can do with a customer, the Marketing-on-Hold Bot can do.

Because the Transport Session can have arbitrarily many channels carrying arbitrarily many media content types, there is considerable flexibility in how the Marketing-on-Hold Bot can be used.

The structure of the Marketing-on-Hold Bot is shown in more detail in Figure 2. The Bot has joined the session using the Communication and Transport Session APIs. For each media type supported by the customer's desktop, a media content handler is responsible for media of that type, accessing media from a content repository which would typically be a collection of media files on hard disk. Four representative content handlers are shown, for audio, video, chat and 3D content. These in turn are synchronized and prompted by a Script Interpreter that runs the marketing script for that customer. The marketing script could be compiled on the fly using past customer history to provide content of interest to that customer.

Each media content type is transmitted to the customer using the Transport Session API, which provides a media channel for each content type. The customer desktop extracts the media content from each channel and outputs it to the customer via a specific media client.

Solution 2 offers the greatest flexibility and the ability to provide interactive content. For example, a Bot with an appropriate knowledge base could respond to queries regarding products, services or any other subject it had been equipped to deal with.

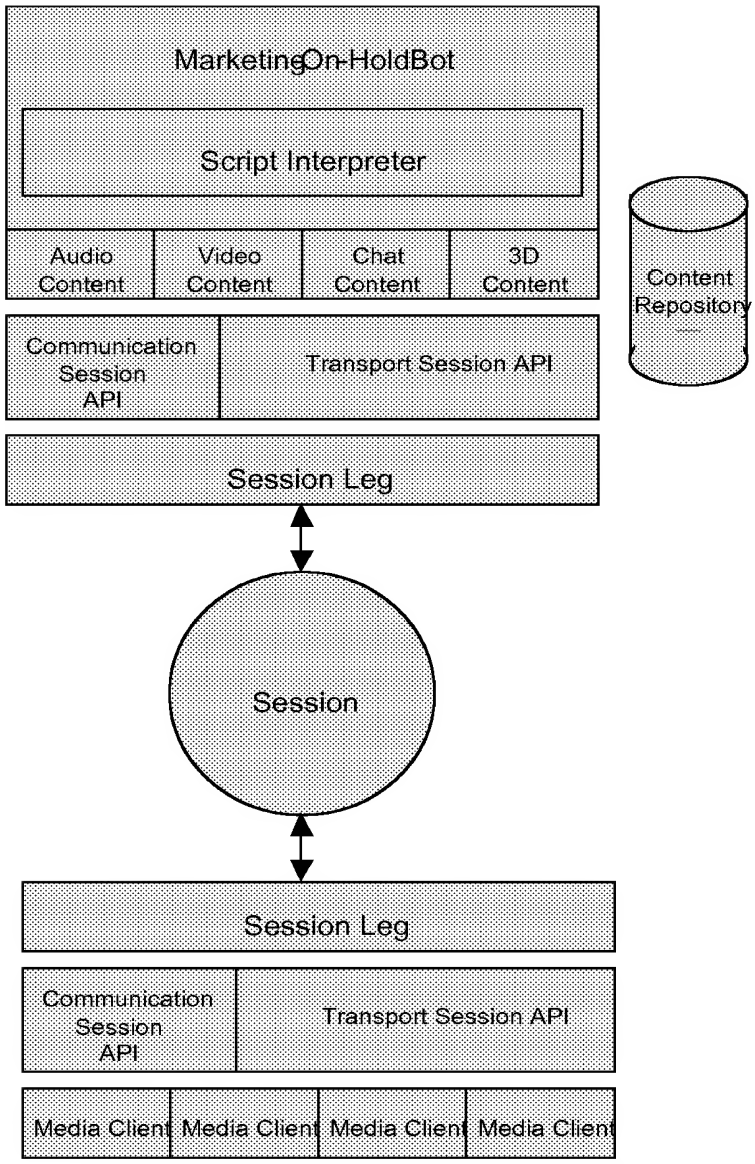


Figure 2. Marketing-On-Hold Bot Architecture

Prior art: Broadvision?

A method of monitoring communication channel content within a call centre and alerting a supervisor when predefined content is detected.

And

A method of collecting data on the performance of a CSR through statistical analysis of the content sent between customer(s) and CSR..

Inventors: Three Wise Men + Gary Crosby (statistical analysis)

Problem

In telephony-based call centers, supervisors monitor calls in order to ensure a certain level of service. This includes ensuring that neither CSR nor customer is on the receiving end of any abuse; the CSR is deterred from unnecessarily extending the length of the call; and the CSRs conduct themselves in a professional manner, among other reasons.

Traditionally this is achieved by walking around the call centre and monitoring telephony calls. This monitoring is usually done silently, but a supervisor may join an existing call at any time and interrupt either party. When multiple media channels are used per call and there is the facility for a CSR to handle multiple calls simultaneously then this becomes an inefficient way to monitor calls. For example, chat is quite a slow conversational medium in comparison with voice and it would be easy for a CSR to unnecessarily extend the duration of the call.

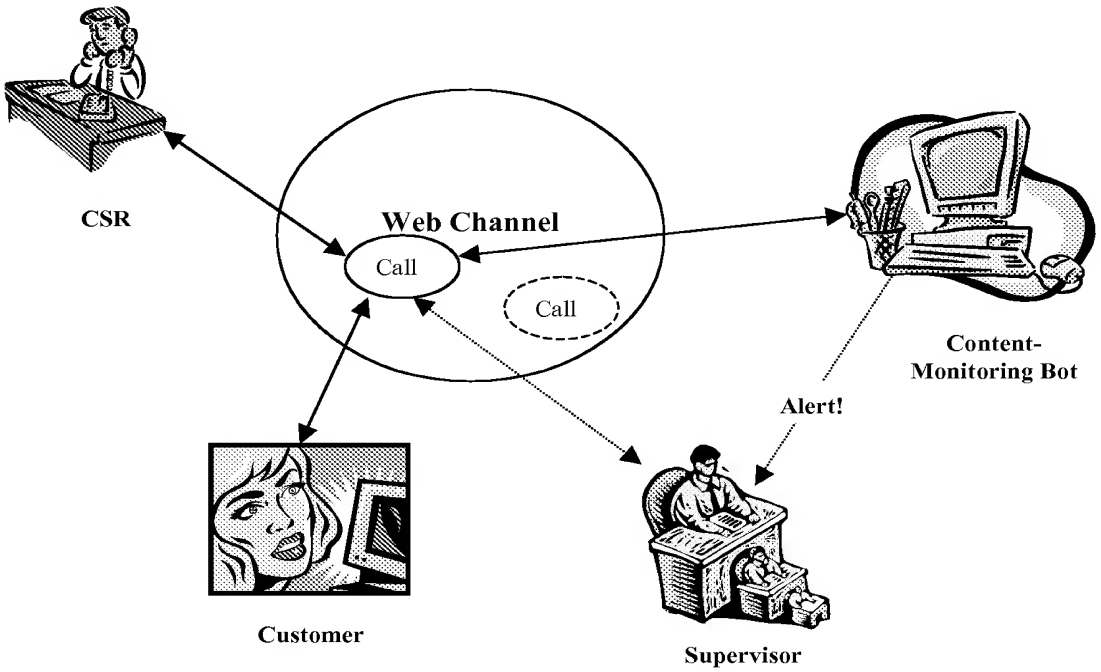


Figure 3. CSR, Customer, Bot, Supervisor Relationships

A content monitoring Bot could be used to help the supervisor in their job. This Bot would match content against a prescribed filter expression and alert the interested parties when the expression is true. Example uses of such filters are:

1. Detecting the presence of profanity or abusive terms and phrases.
2. Detect the transmission/sharing of undesirable material or references to such material.

The Bot would also be in a position to collect statistics on the call which could be used to identify (among others):

1. Call duration.
2. How quick the CSR was in responding to each of the customer's inputs.
3. Calls that had a high response and long duration indicating a talkative CSR.

There are other applications of the content monitoring Bot, such as using the chat transcripts for refining the Frequently Asked Questions knowledgebase found on most web sites, acting as a "did you know?" helper that suggests other material that might be relevant when it detects keywords, etc., but they are not covered in these inventions.

Figure 3 shows the how the Bot fits into the standard communication session.

Method

The Content-Monitoring Bot is a software package that joins a session silently (nobody can tell that it is present) using the same interfaces as the customer and CSR desktops. These interfaces are used to monitor all media content exchange between customers and CSRs. All the media clients (see section 6.5) in the customer desktop are exposed in this way. Anything a CSR or customer can do, the Content-Monitoring Bot can do too.

Because the Transport Session can have arbitrarily many channels carrying arbitrarily many media content types, there is considerable flexibility in how the Content-Monitoring Bot can be used.

Each media content type is transmitted to the customer using the Transport Session API, which provides a media channel for each content type. The customer desktop extracts the media content from each channel and outputs it to the customer via a specific media client.

The structure of the Content-Monitoring Bot is shown in more detail in Figure 4. The Bot has joined the session using the Communication and Transport Session APIs. For each media type supported by the customer's desktop, a media content handler is responsible for media of that type. Four representative content handlers are shown, for audio, video, chat and 3D content.

The Filter Manager interfaces with those media handlers to evaluate the filters it has been provided with against incoming content.

Should any filter match some of the content, then the Alert Manager is informed and an appropriate action is taken depending upon which filter matched and what the media channel was. Typically that action is to send a message to the supervisor's desktop indicating what the problem is, the call details and the offending content. The supervisor can then decide whether any further action should be taken.

The Stats Manager uses the media handler interfaces to extract pertinent statistical information that is logged to a database for later evaluation.

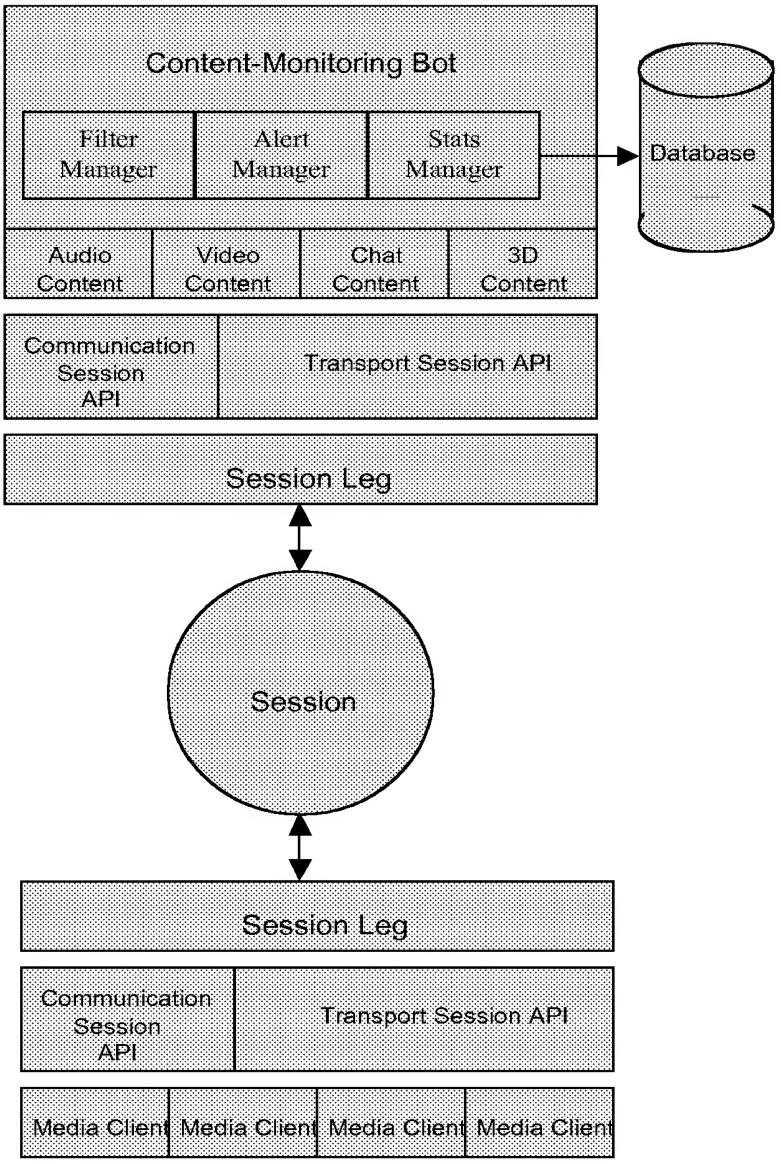


Figure 4. Content-Monitoring Bot Architecture.

The application of the Content-Monitoring Bot described here is within a call centre, but it is applicable to multimedia communications in general and specifically, interaction-enabled web sites.

A method for managing multiple multi-channel communication sessions on a single desktop by coordinating overloaded, channel-specific, media displays with a single operation.

Inventors: Three Amigos

Problem

Office workers deal with multiple communication sessions using different media every day, e.g. phone, fax, email, text chat, pagers, SMS, collaborative web browsing, etc. Each media has it's own interface and a number of functions that can be performed, some of them common to all media, some unique. Handling multiple media for a single call places great demands on a desktop real estate, e.g. imagine requiring a web browser, email client and database application to address a single call. Add to this the requirement that multiple calls can be handled simultaneously and it becomes clear that things can quickly spiral out of control. In addition, each application will be performing their own session management functions, e.g. joining, leaving, etc., which would be better administered centrally, thus reducing code size, maintainability, etc.

This invention presents a mechanism for coordinating the session management of the many channels that can be utilised when dealing with interaction between two or more people; specifically, in a call centre environment where those calls are between CSR and customer(s) over the web.

Method

The essence of the solution is to centralise functions that each media channel must perform, provide a one-stop reference point to determine the status of each session, and multiplex the user interfaces so the same amount of desk space is used regardless of how many sessions are being handled simultaneously. This arrangement will be termed a Multiple Session Management Interface (MSMI) and also has the important property that the amount of information a user must assimilate at one time is minimised.

MSMI handles the functions common to these media, e.g. joining and leaving a communications session. Individual media channel interfaces provide access to the functions specific to that media type, acting as displays for its content and providing a terminal for input. However, it is also acceptable to have a single user interface for multiple channels.

To achieve this MSMI has the following functions:

1. All pertinent information associated with each session is displayed to help user distinguish between them, e.g. name of person(s) in session, name of session, status of session, etc.
2. Ability to switch between multiple sessions with minimal fuss.
3. Clear notification when an invitation to join a new session is received.
4. Notification when content of a session other than the currently selected one has changed.
5. Indication whether the associated channel is being used in the selected session.
6. Provide a record of content exchanged on the channel for every session (the point at which the record is deleted is implementation dependent).

Figure 5 shows the relationship between the low-level media handlers (one per channel in any given session) and the user interface elements. There may be any number of channels in a session.

Not all sessions utilise the same media types, e.g. in the diagram media type 2 is only used by sessions A and B. Actual implementation screen shots are available in section 6.3 of "A Web Interaction System".

Functions 1-4 are handled by the Centralised Session Control component, functions 5 & 6 being handled by each of the user interfaces for the media channels.

The Centralised Session Control window contains a table with each row representing a session that the user is invited to, is currently involved in, or has recently left. The row displays contact information associated with the session. New invitations are highlighted in red, the currently selected session is highlighted in blue. Associated with this main window are a number of other windows, one per media channel client. As different sessions are selected, so the media windows show the content of the channel specific to that call (if used). Also, each user interface function is disabled or enabled depending upon the status of the currently selected session to ensure that the user is only presented with the options relevant at any given moment. Satisfies functions 1-3.

When content arrives on a media channel for a call other than the one currently selected, an icon representing that media type appears in the table row associated with the session. When the user selects a session, the media icons are cleared. Satisfies function 4.

If a media channel is not being used in a session then the associated user interface indicates this to the user, but the interface does not disappear from sight.

When the user leaves the session, the session's details remain in the table and when selected, the transcripts for each of the media channels are available from the respective GUI windows. When the session is removed from the table, the transcripts are deleted. Satisfies function 6.

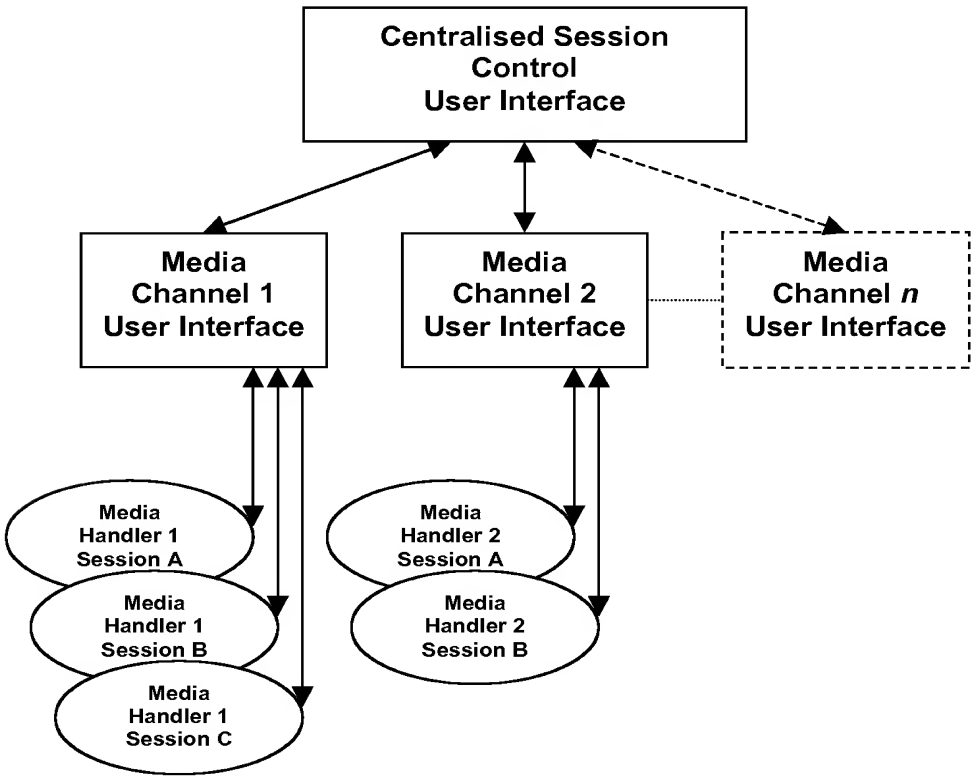


Figure 5. Relationship between media handlers for each session and the user interface.

This invention has concentrated on the application of the MSMI to media management, but the interface exposed by each Media Channel User Interface and the Centralised Session Control User Interface is quite simple and third-party applications, e.g. database front-ends, could be wrapped in code implementing this interface. This would then provide complete synchronisation of all desktop applications related to a specific session (call).

A mechanism for supporting multi-locale web-based communication sessions by dynamic configuration of the user interface.

Inventors: Three Tenors

Problem

In a telephony-based call centre, it is quite straight forward for a multi-lingual CSR to handle calls in different languages and dialects since all communication is by voice. When other media are used, e.g. test chat, web content, etc., then there is a requirement that the CSR desktop can receive and transmit content in the appropriate *locale* (language + country of origin), with the appropriate conventions.

Building upon the MSMI invention, this invention describes how the MSMI can be extended to support multiple simultaneous calls in different locales, potentially originating from different countries.

Method

Associating a session with a specific locale can be done at any stage prior to the creation of the media channel user interfaces. In the Web Interaction System, the locale can be embedded in the web content the customer is perusing as they request help. This can be done by associating a locale with a help button that appears on the web site (see embedding contact context in URL patent). If the web-site is split into language specific regions, then there is only need for one help button on a page. Alternatively, if there are no specific regions in the site, then multiple buttons can be provided, each one coded for a different locale. Of course, this information could just be provided as part of a form-based dialogue after the help button was pressed. The choice is in the hands of the web-site designer.

The locale information is passed via the Session Mediation Server (see Figure 1) to the Communications Session Manager and is finally presented as part of the session details to the CSR desktop. If the CSR chooses to accept the call, then this information is saved by the Centralised Session Control component (Figure 5).

Every CSR desktop is configured to operate in either the chosen default locale of the call centre, or the CSR's native language. The Centralised Session Control user interface always uses this locale, regardless of the locale used in the selected session. Those parts of the media channel user interfaces that are not session-specific are also always presented in the desktop locale. Those parts that are session-specific are reconfigured dynamically to the correct locale as each session is selected. Session-specific parts include, character sets, conventions, e.g. date, time, etc., scripts or other pre-defined content intended to be pulled in during the session, and so on.

Pre-defined, locale-dependent content, e.g. scripts, can be pulled in at run-time if the CSR desktop receives a session from a locale that it does not have the appropriate version for.

A method for interleaving multimedia scripts with user input to individual media displays.

Inventors: Three Blind Mice

Problem

A CSR will typically answer the same questions over and over again. In a telephony-based call centre there is no option other than to repeat the answer, but when using web-based communication technologies, e.g. text chat, there is the opportunity to place these commonly used answers/pieces of information into scripts. This concept can also be extended to authoring web-site tours which can be delivered using text chat and page push in synchrony.

Synchronised multimedia scripts are not a new idea: Synchronized Multimedia Integration Language (SMIL - http://www.w3.org/TR/REC-smil) provides the facility to author multimedia scripts. Standard SMIL players output all media content on the same display. This invention is based around the fact that each media may be output to different displays and that the scripts are used as a whole or piecemeal basis interleaved with CSR input and driven from any media channel user interface.

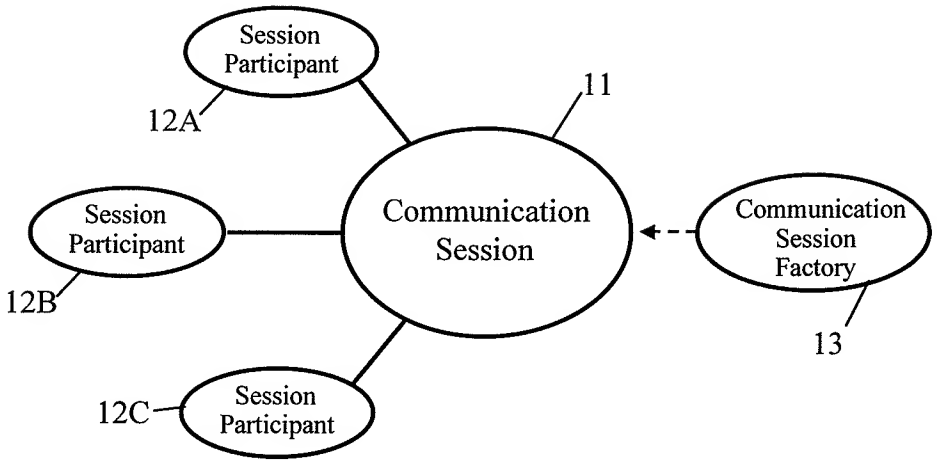
Method

Scripts are loaded by the Centralised Session Control component (see Figure 5) and accessed by each media channel user interface. Each script is broken down into a number of entry points that correspond to items on pull-down menus available from the media channel UIs. An entry point may only be available from one media channel, or the same entry point may be available from all media channel UIs whose content is involved in that script entry.

Using such a facility it is not only easy to build up scripts of common dialogue that a CSR may use each day (and quicker to deploy than typing), but to coordinate common actions such as typing "let's look at printer model X" in the text chat window and then pushing the relevant URL in the page push channel. A single script menu entry in the chat media UI can trigger both actions.

More complex sequences of actions can be coordinated within scripts with the ability to pause and restart them. This ensures that the CSR can, at any time, add comments relevant to the call they are dealing with, personalising the experience for the customer.

Needs more detail.



Fiaure 1

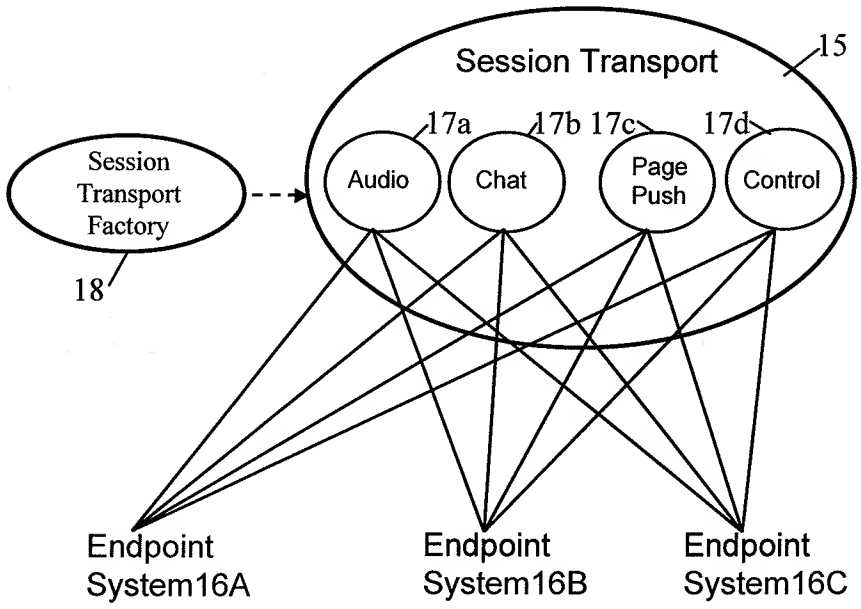


Figure 2

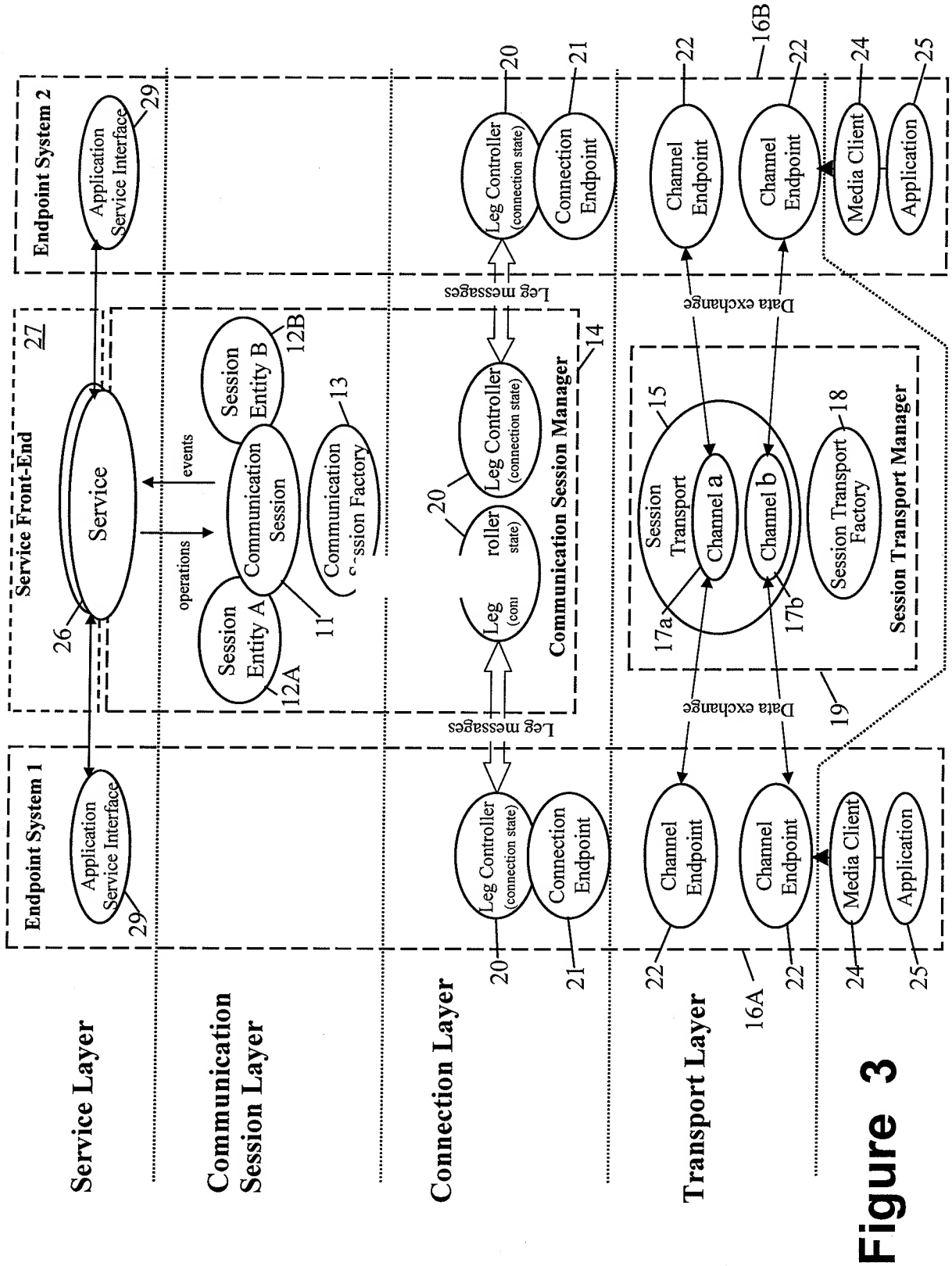

Service Layer
Communication Session Layer
Connection Layer
Transport Layer
Endpoint System 1
Endpoint System 2
Application Service Interface
29
Service Front-End
27
Service
26
operations
events
Session Entity A
12A
Communication Session
11
Session Entity B
12B
Communication Session Factory
13
Communication Session Manager
14
Leg Controller (connection state)
20
Connection Endpoint
21
Leg messages
Session Transport Manager
19
Session Transport
15
Channel a
17a
Channel b
17b
Session Transport Factory
18
Channel Endpoint
22
16A
16B
Data exchange
Media Client
24
Application
25

Figure 3

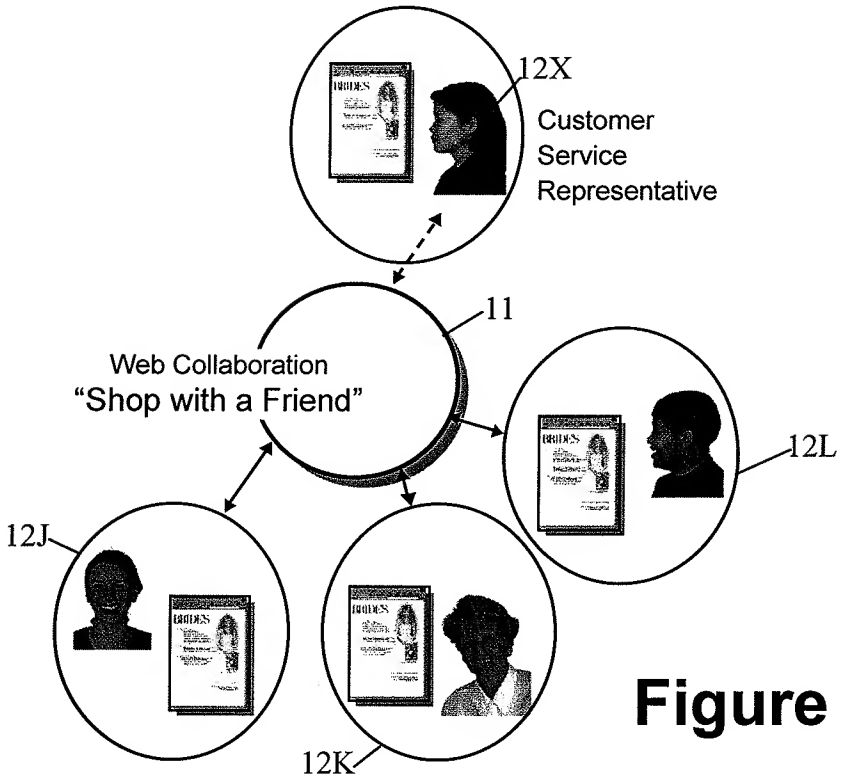


Figure 4

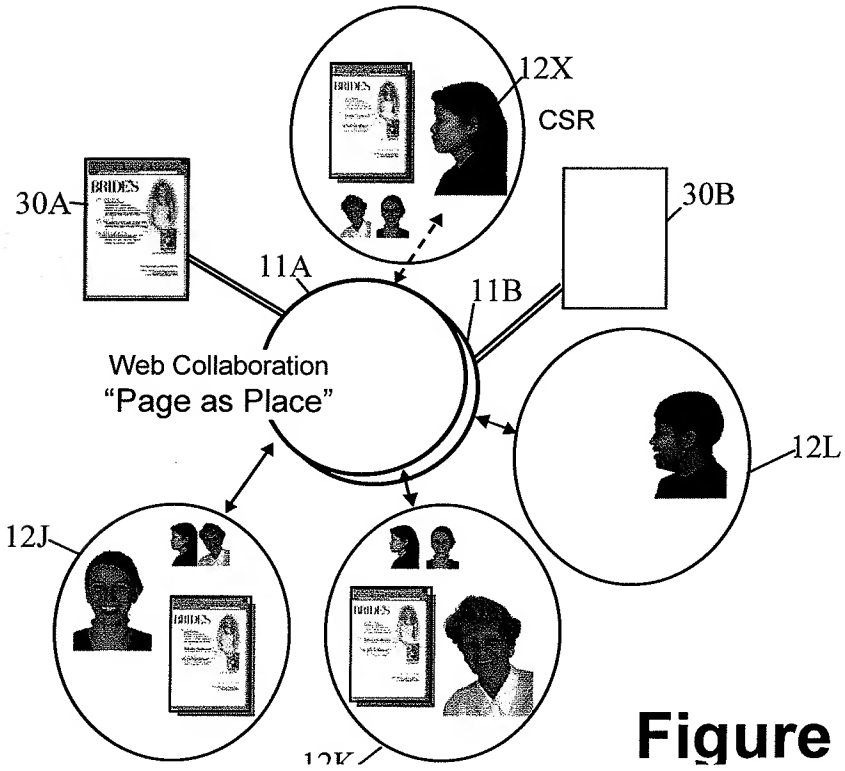


Figure 5

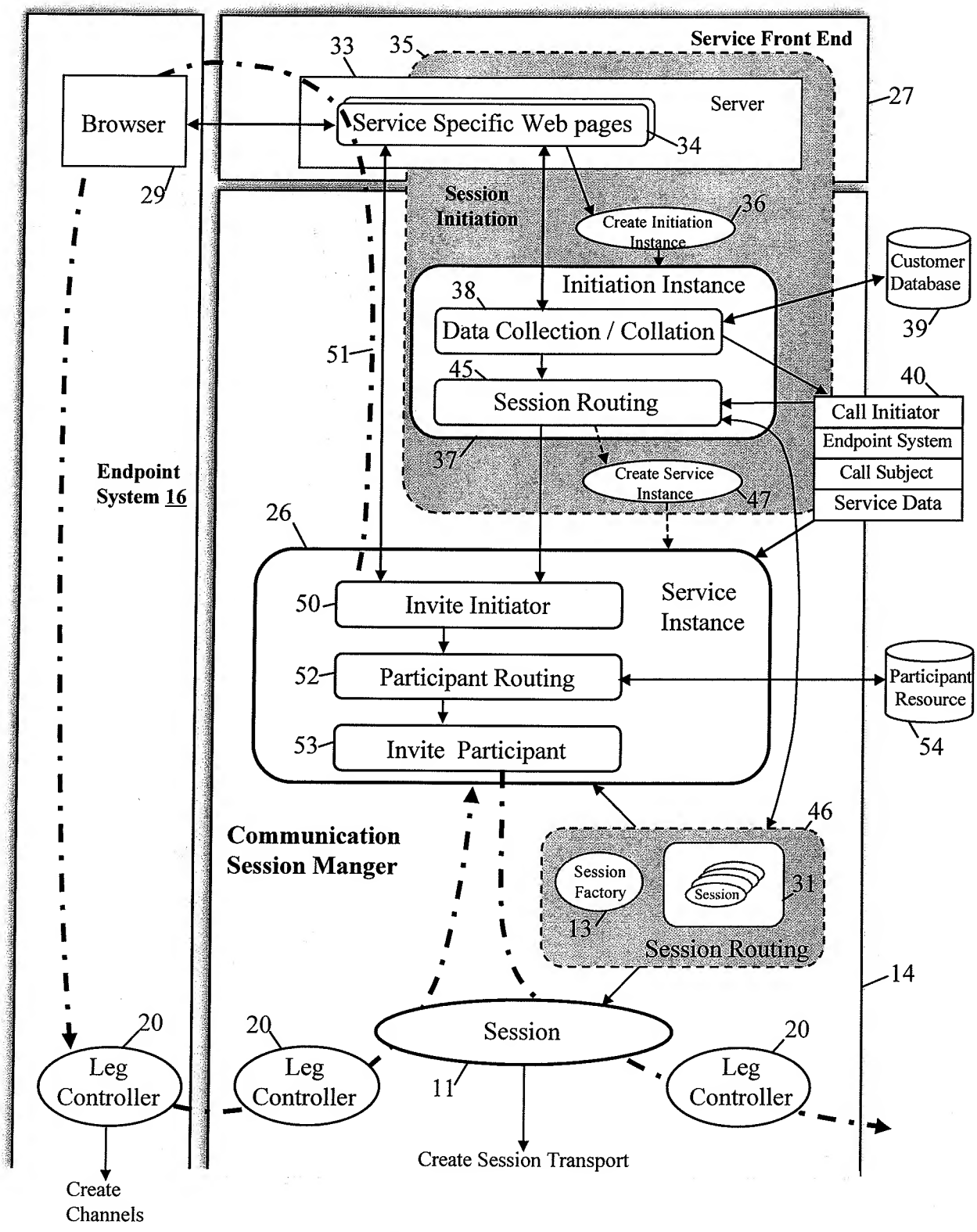

Service Front End
Server
27
33
35
Browser
29
Service Specific Web pages
34
Session Initiation
Create Initiation Instance
36
Customer Database
39
Initiation Instance
38
Data Collection / Collation
45
Session Routing
37
51
40
Call Initiator
Endpoint System
Call Subject
Service Data
Create Service Instance
47
Endpoint System 16
26
Service Instance
50
Invite Initiator
52
Participant Routing
53
Invite Participant
Participant Resource
54
Communication Session Manger
46
Session Factory
13
Session
31
Session Routing
14
20
Leg Controller
Session
11
Create Session Transport
Create Channels

Figure 6

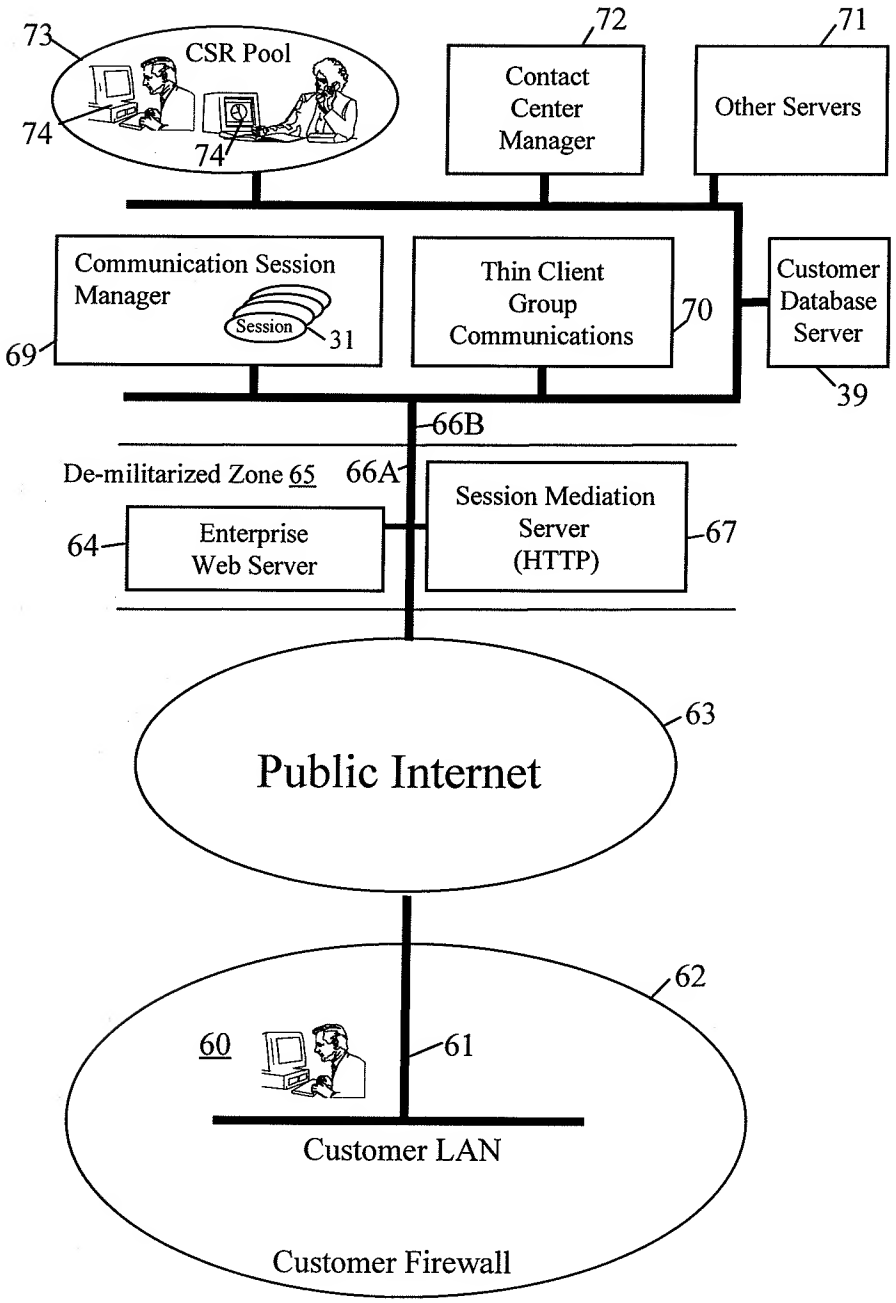

73
CSR Pool
74
74
72
Contact Center Manager
71
Other Servers
Communication Session Manager
Session
31
69
Thin Client Group Communications
70
Customer Database Server
39
66B
De-militarized Zone 65
66A
Session Mediation Server (HTTP)
67
64
Enterprise Web Server
63
Public Internet
62
60
61
Customer LAN
Customer Firewall

Figure 7

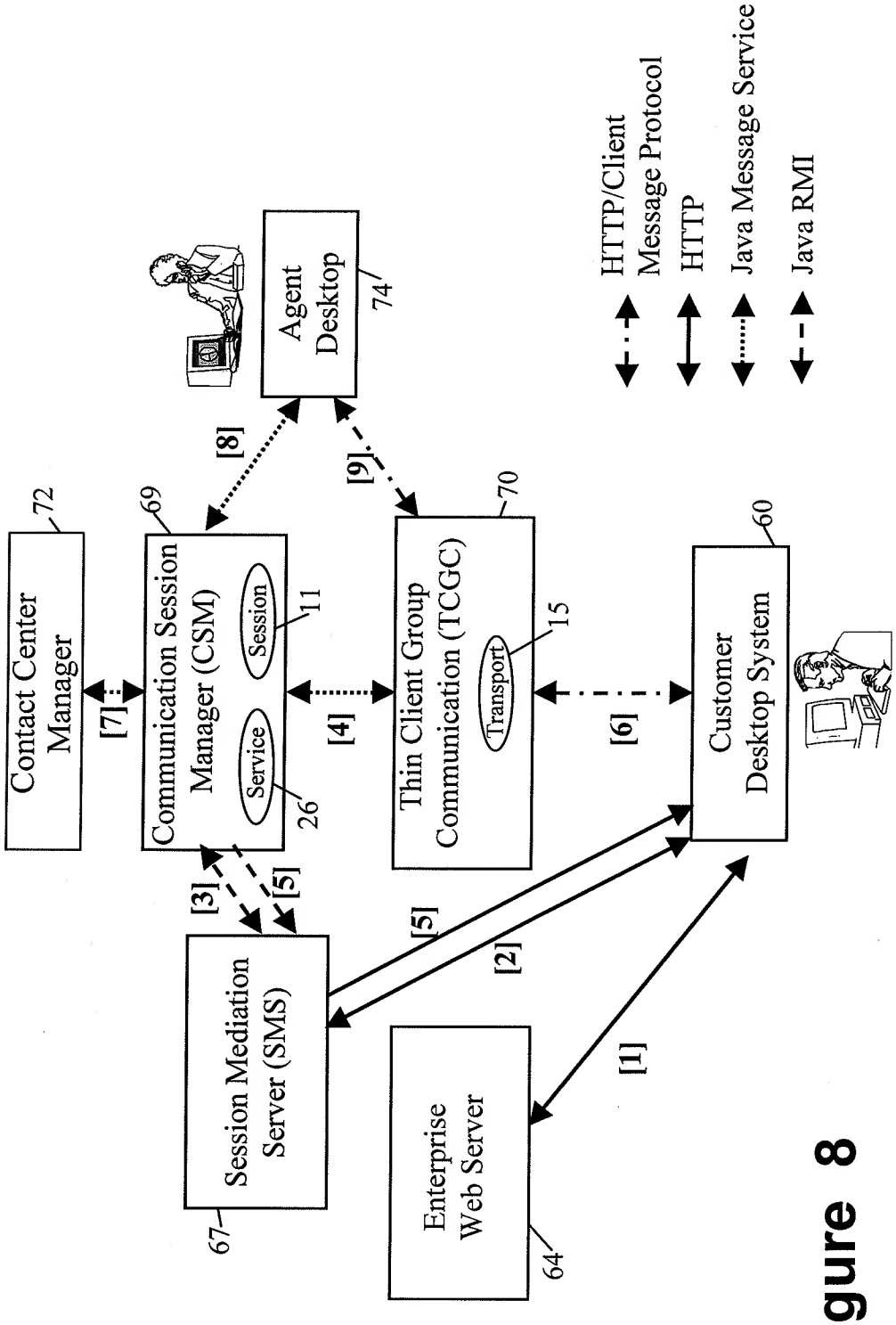

Contact Center Manager
72
Communication Session Manager (CSM)
69
Session
11
Service
26
Agent Desktop
74
Thin Client Group Communication (TCGC)
70
Transport
15
Customer Desktop System
60
Session Mediation Server (SMS)
67
Enterprise Web Server
64
[1]
[2]
[3]
[4]
[5]
[5]
[6]
[7]
[8]
[9]
HTTP/Client Message Protocol
HTTP
Java Message Service
Java RMI

Figure 8

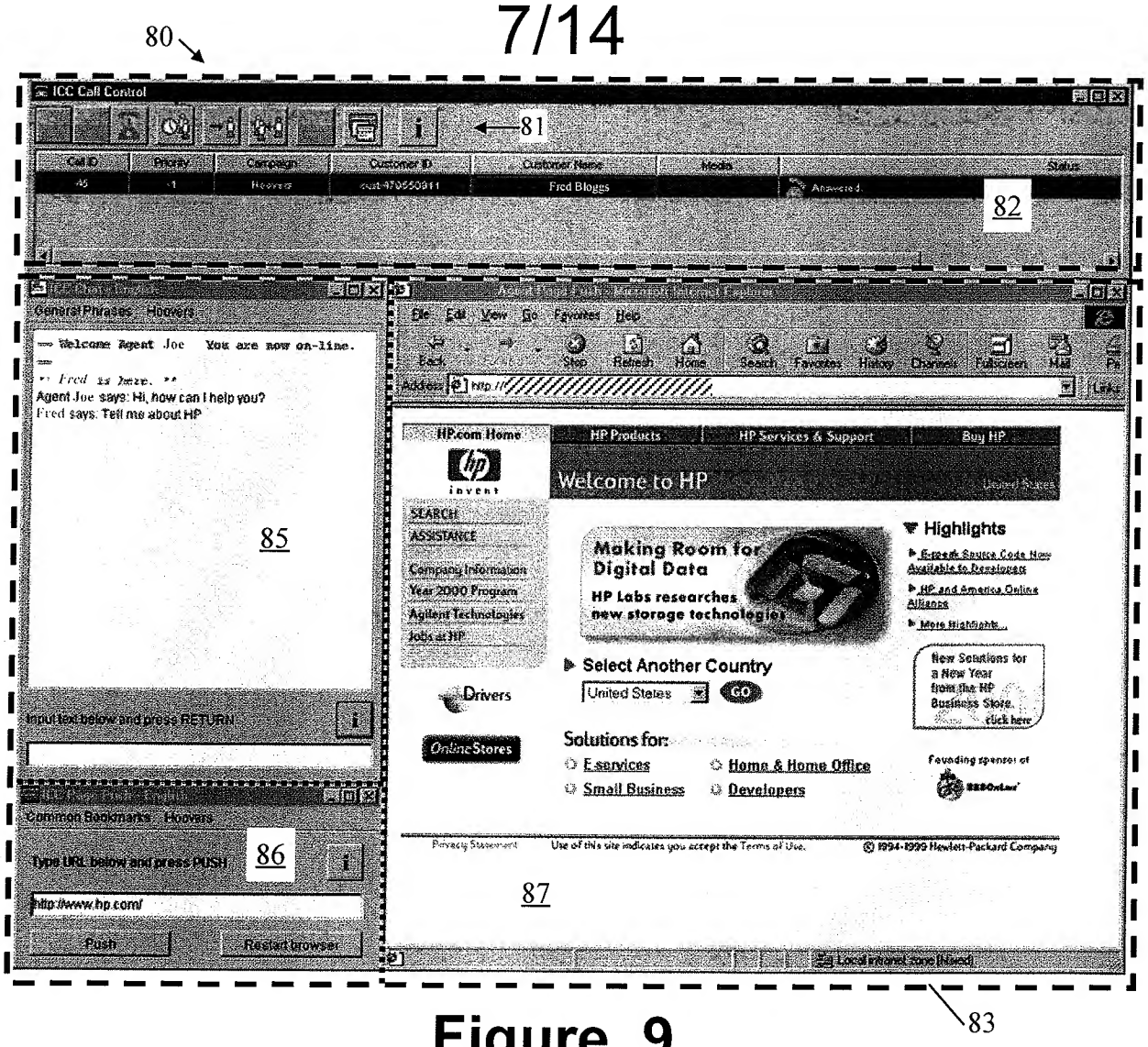


Figure 9

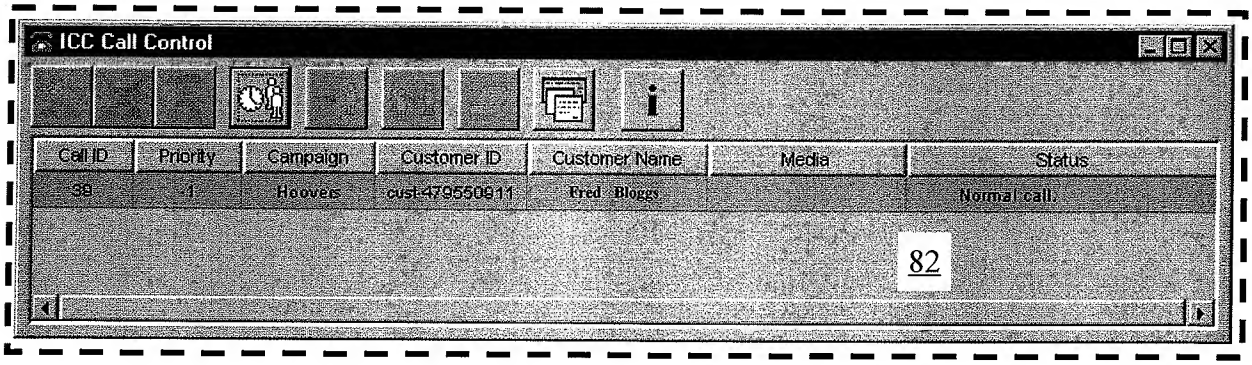


Figure 14

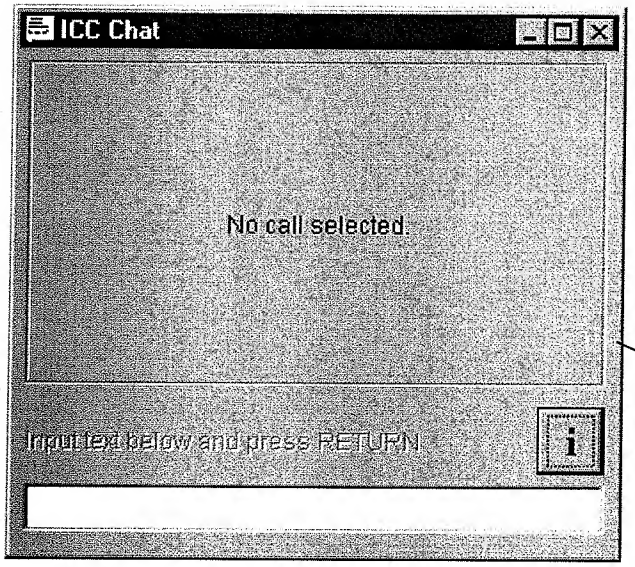


Figure 10

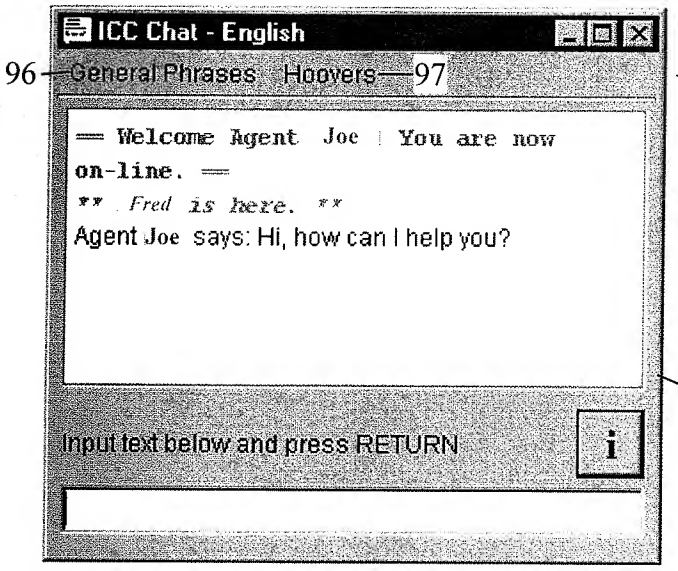


Figure 11

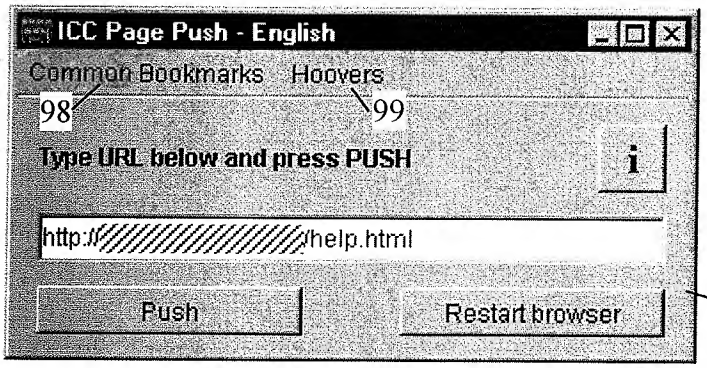


Figure 12

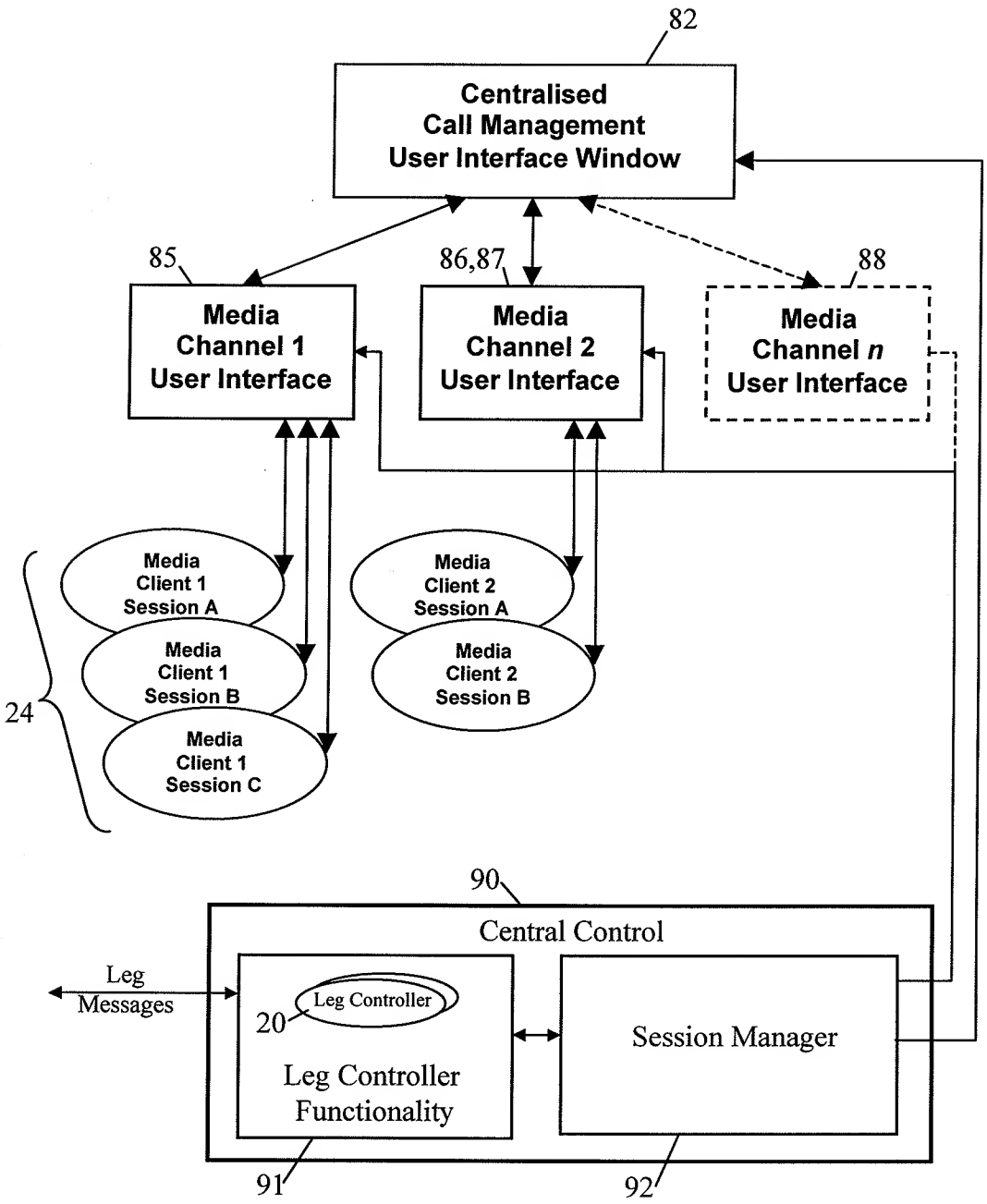

82
Centralised
Call Management
User Interface Window
85
86,87
88
Media
Channel 1
User Interface
Media
Channel 2
User Interface
Media
Channel n
User Interface
Media
Client 1
Session A
Media
Client 1
Session B
Media
Client 1
Session C
Media
Client 2
Session A
Media
Client 2
Session B
24
90
Central Control
Leg
Messages
Leg Controller
20
Leg Controller
Functionality
Session Manager
91
92

Figure 13

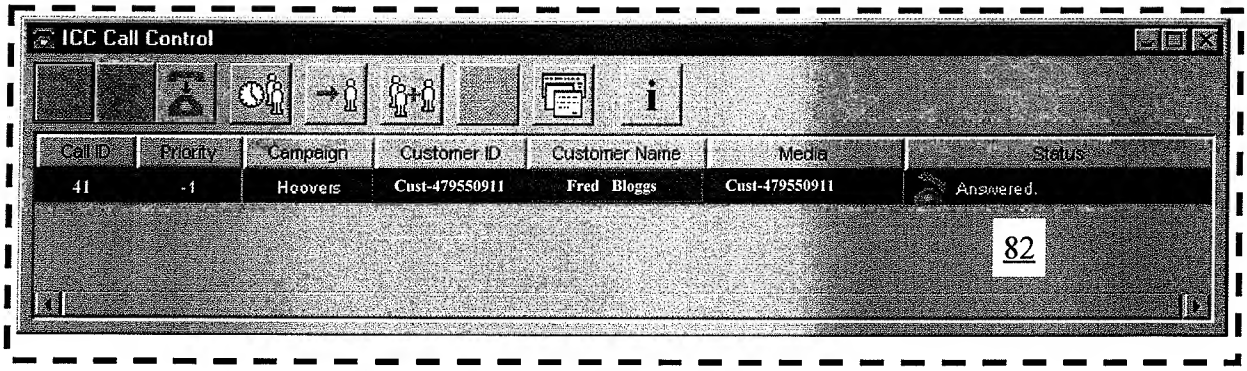


Figure 15

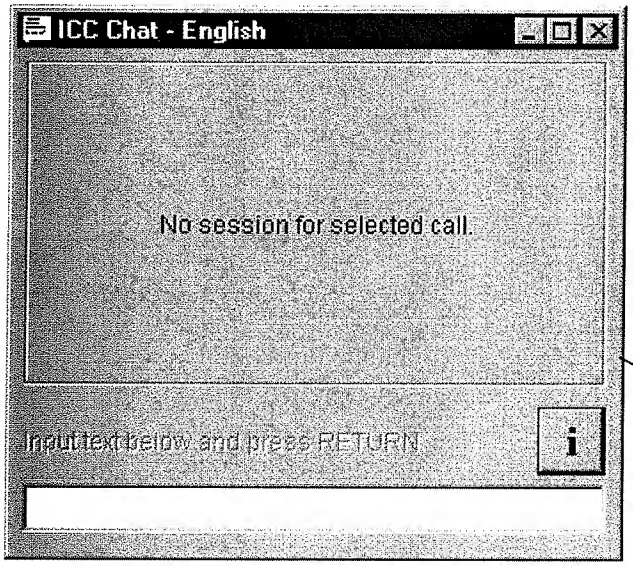


Figure 16

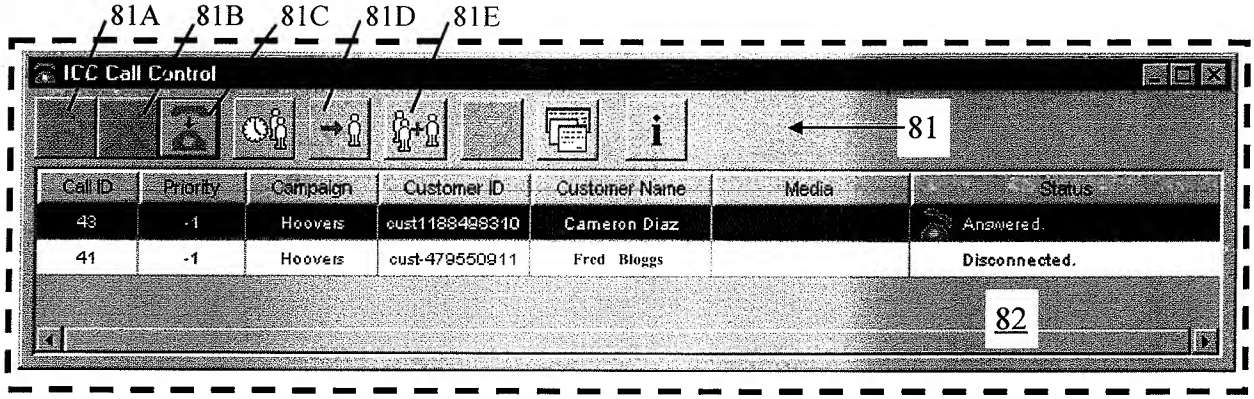


Figure 17

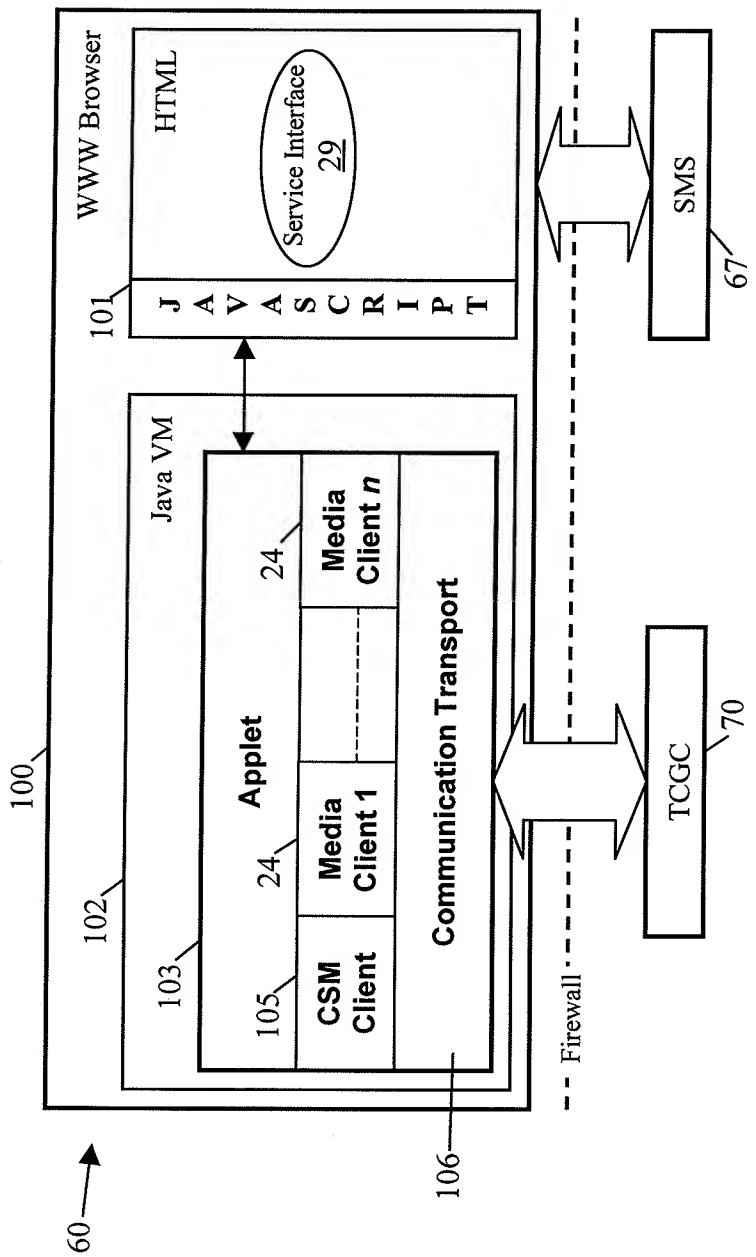

WWW Browser
HTML
Service Interface
29
JAVASCRIPT
101
Java VM
Applet
24
Media Client n
Media Client 1
CSM Client
Communication Transport
105
103
102
100
106
60
Firewall
SMS
67
TCGC
70

Figure 18

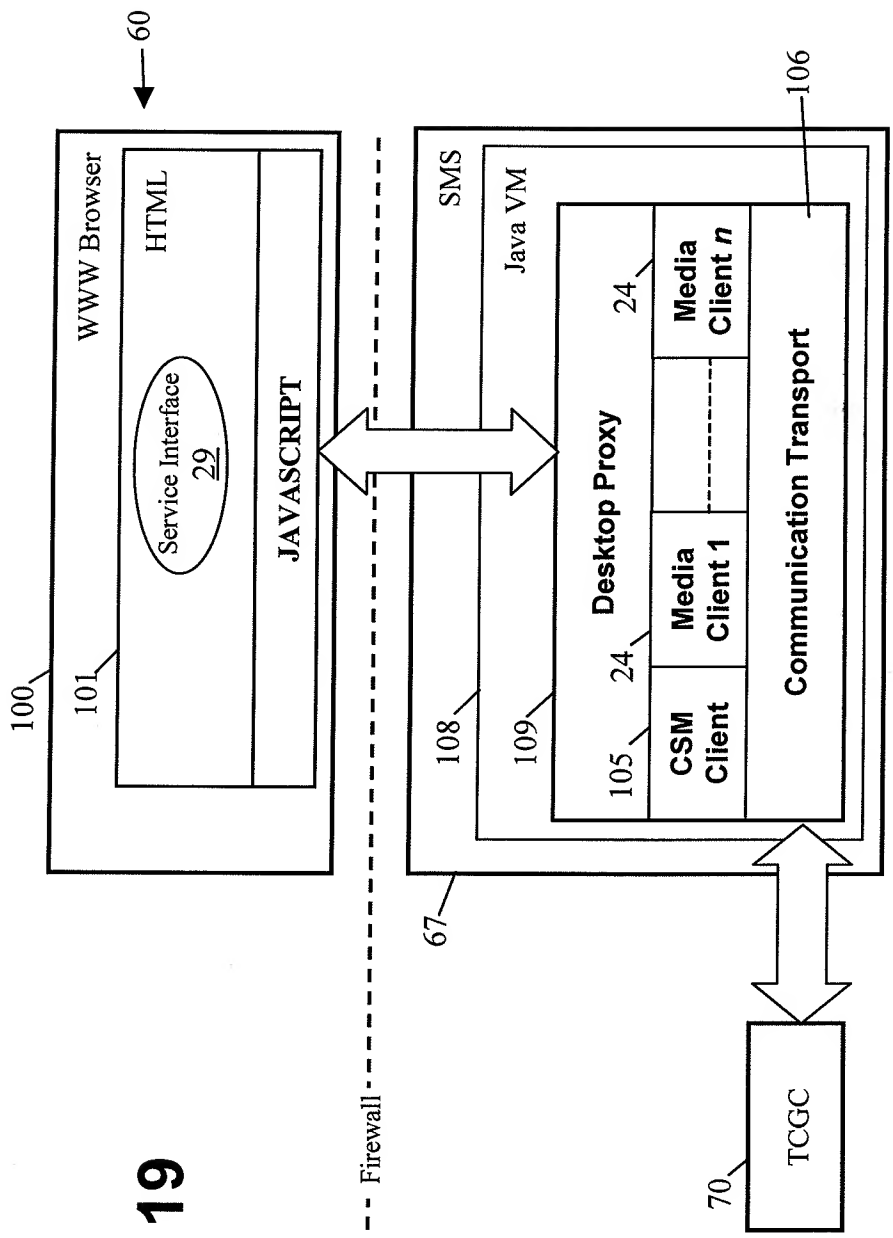

60
WWW Browser
HTML
Service Interface
29
JAVASCRIPT
100
101
Firewall
SMS
Java VM
Desktop Proxy
24
Media
Client n
Media
Client 1
CSM
Client
105
108
109
106
67
TCGC
70

Figure 19

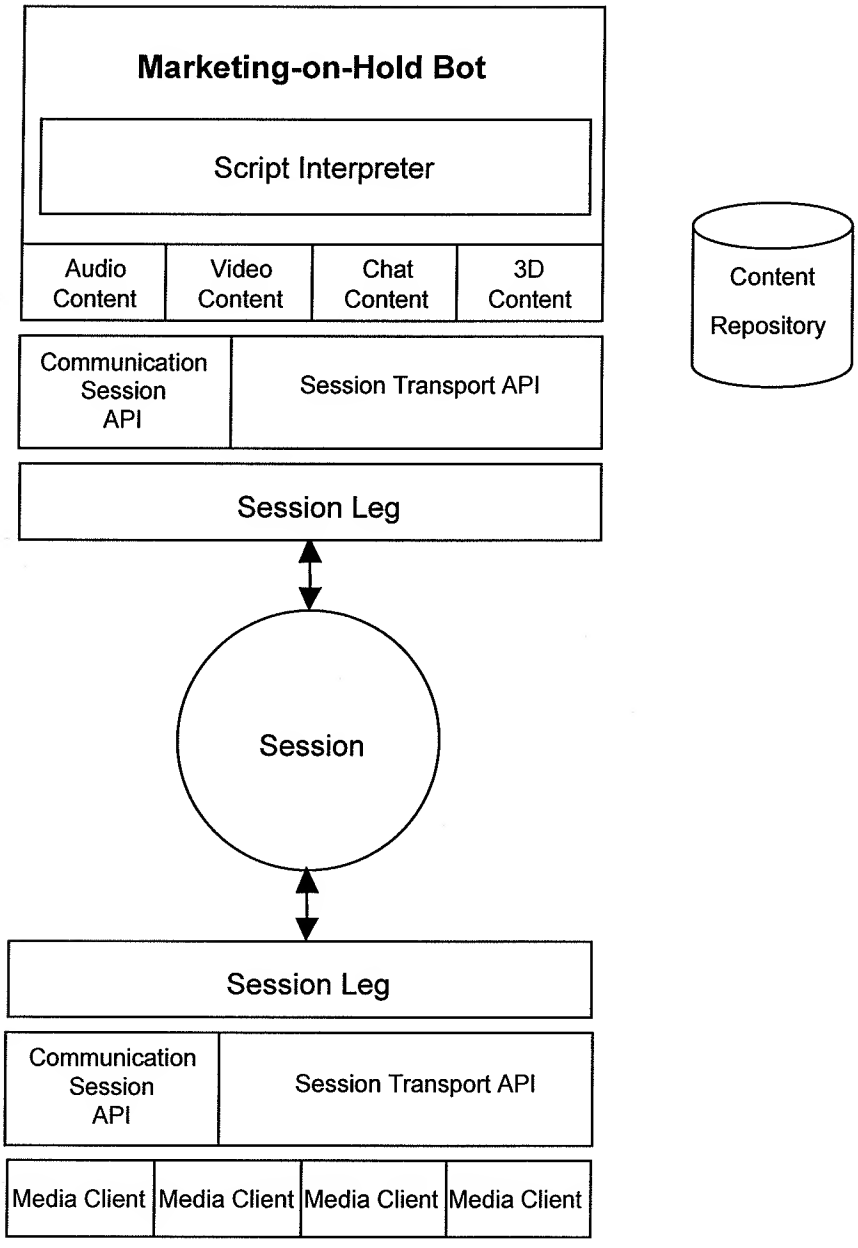

Marketing-on-Hold Bot
Script Interpreter
Audio Content
Video Content
Chat Content
3D Content
Content Repository
Communication Session API
Session Transport API
Session Leg
Session
Session Leg
Communication Session API
Session Transport API
Media Client
Media Client
Media Client
Media Client

Figure 20

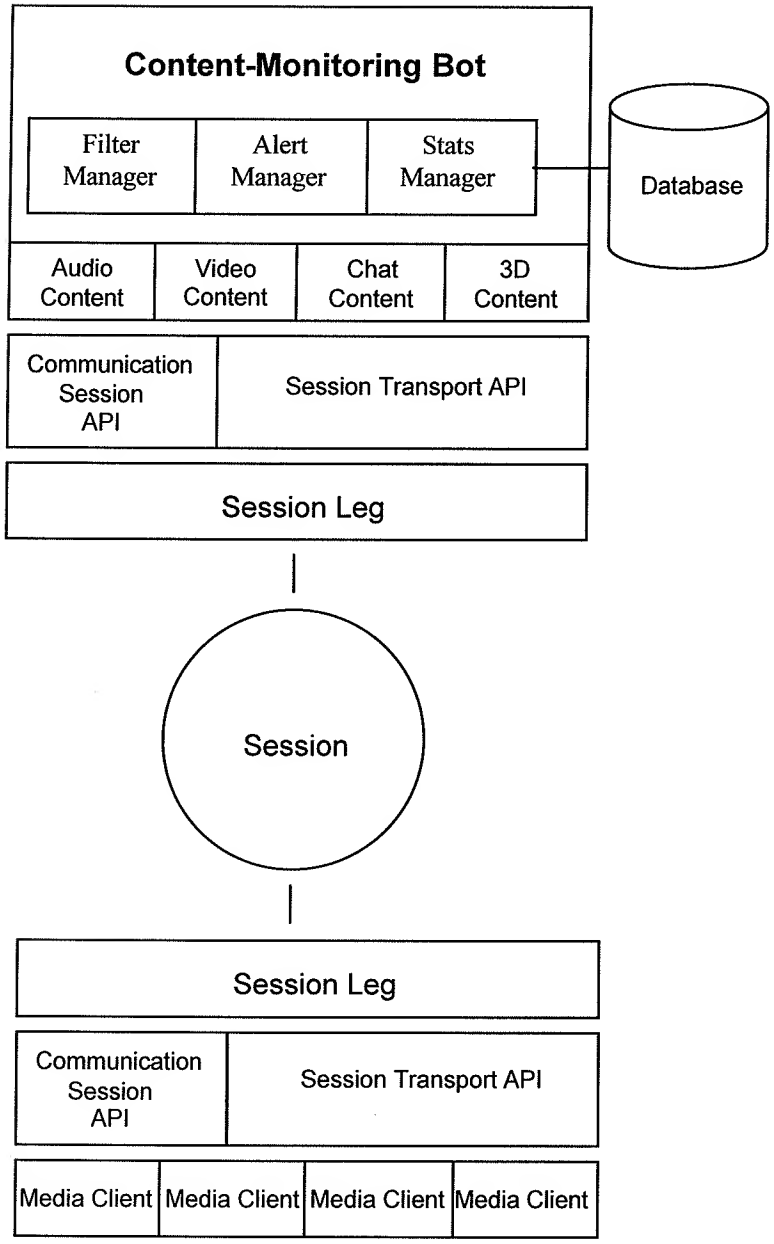

Content-Monitoring Bot
Filter Manager
Alert Manager
Stats Manager
Database
Audio Content
Video Content
Chat Content
3D Content
Communication Session API
Session Transport API
Session Leg
Session
Session Leg
Communication Session API
Session Transport API
Media Client
Media Client
Media Client
Media Client

Figure 21

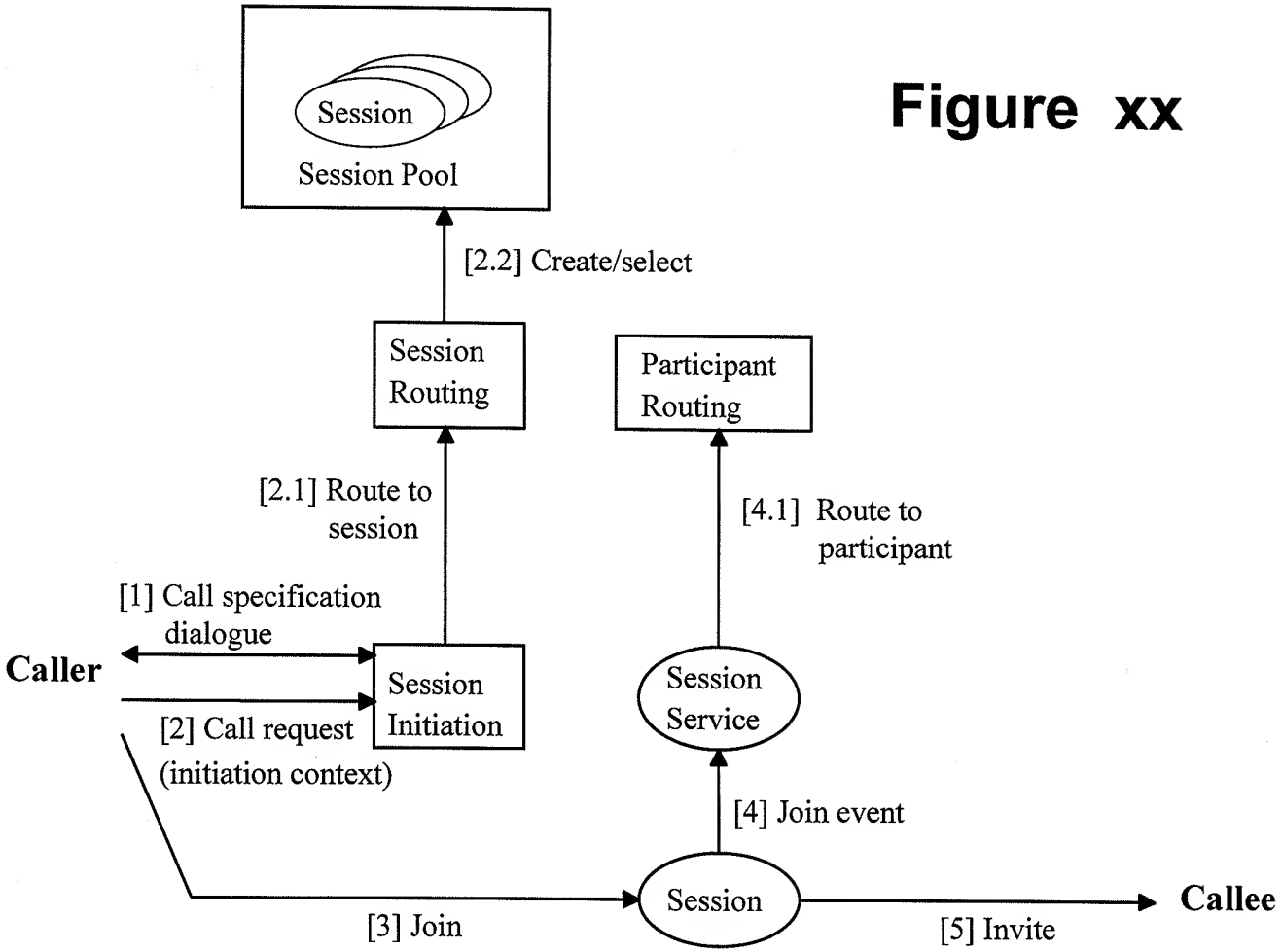

Session
Session Pool
[2.2] Create/select
Session Routing
Participant Routing
[2.1] Route to session
[4.1] Route to participant
[1] Call specification dialogue
Caller
Session Initiation
Session Service
[2] Call request (initiation context)
[4] Join event
Session
Callee
[3] Join
[5] Invite

Figure xx

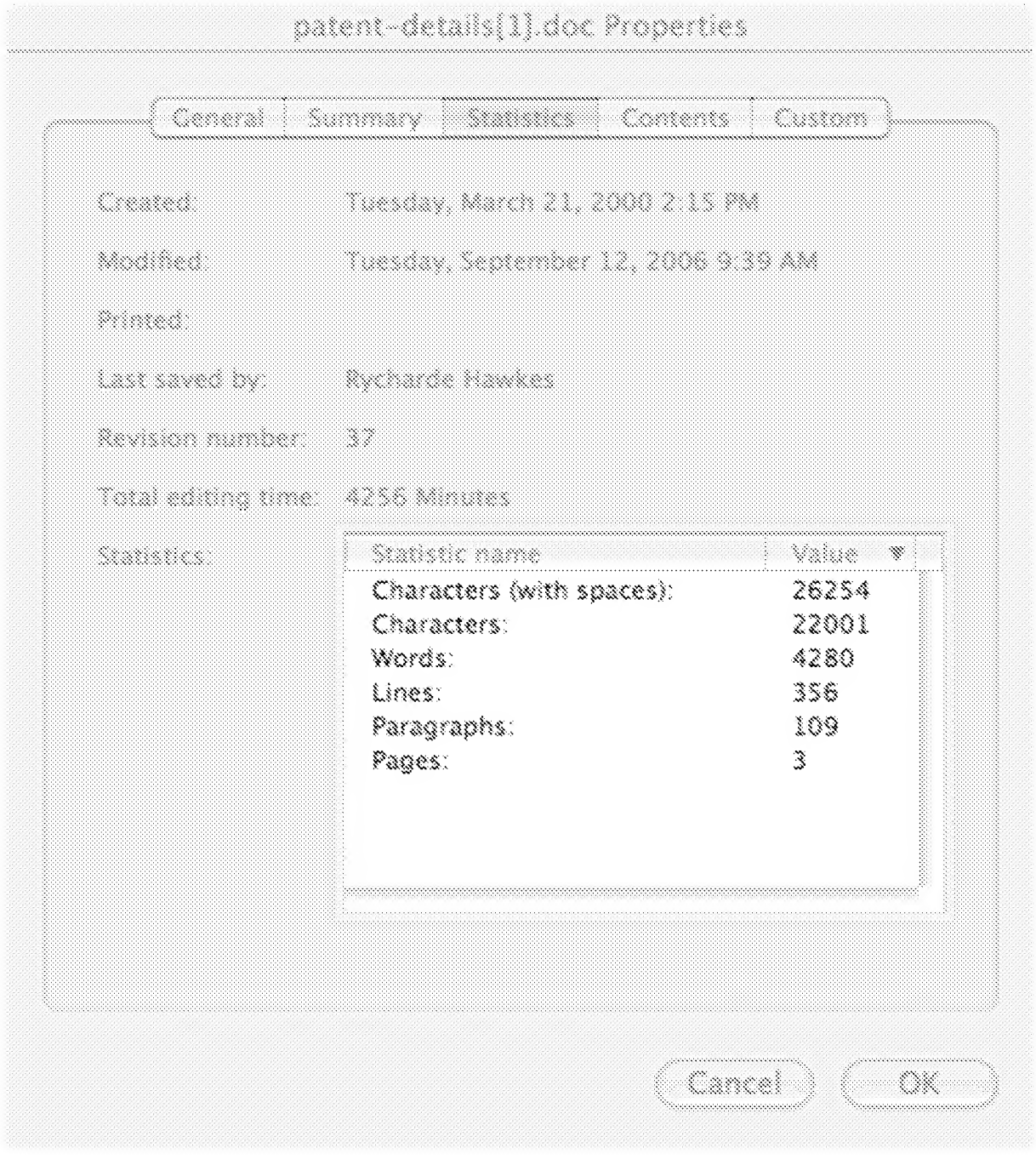

patent-details[1].doc Properties
General
Summary
Statistics
Contents
Custom
Created: Tuesday, March 21, 2000 2:15 PM
Modified: Tuesday, September 12, 2006 9:39 AM
Printed:
Last saved by: Rychards Hawkes
Revision number: 37
Total editing time: 4256 Minutes
Statistics:
Statistic name
Value
Characters (with spaces): 26254
Characters: 22001
Words: 4280
Lines: 356
Paragraphs: 109
Pages: 3
Cancel
OK

Exhibit E

Web Interaction System

Field of the Invention

The present invention relates to a web interaction system for collaboration between two or more parties over the World Wide Web. As used herein, the term World Wide Web means a network of systems running applications using the HTTP protocol (any version) and similar and successor protocols.

Background of the Invention

The Internet and the World Wide Web (WWW) have made it possible for enterprises to sell products and services by using the WWW to describe offers, using various means such as WWW forms or electronic mail to conduct transactions. This form of selling is based around the catalogue model that originated in the 19th Century, where the WWW site substitutes for the paper catalogue, and the postal service is replaced by the modern online equivalent.

Many enterprises currently use the telephone to replace or augment the catalogue model. A customer can call the organisation and purchase goods and services interactively over the telephone. This has the advantage that a customer can interact directly with a Customer Service Representative (*CSR* or "*CSR*"), but has the disadvantage that the telephone is a non-visual medium.

The need to handle large numbers of customers simultaneously, and the concurrent need to manage a pool of CSRs, has led to the development of the *call centre*, and the development of specialised software control packages to determine how incoming customer calls are routed to CSRs.

It is possible to combine the catalogue model of WWW selling with the telephone call centre (and other communication channels) to produce what is often called *contact centre*. The contact centre is like a telephone call centre, but instead of CSRs handling only telephone calls, they may be expected to handle customer communications in a variety of formats: FAX, electronic mail, telephone and WWW are typical. A contact centre is characterised by multiple contact or communication channels, and a pool of CSRs who interact with customers to provide services, products or support. The contact centre

provides the illusion of a single point of contact for customers on a regional, national or even international basis.

The online purchasing experience can be further enriched by taking ideas from the familiar mall or high street shopping experience, where the social element of viewing and purchasing goods and services becomes important. It is common for people to go shopping together, to inspect goods and offers together, and to discuss the pros and cons of a given item. Sales assistants can operate in a number of different modes: they can approach a customer who looks in need of help, they may work in a specific area such as cosmetics or cameras, they may work on an enquiry counter, they may provide demonstrations of appliances, or they may be generally available and circulate around different parts of a store. It is common for sales assistants to consult each other and share knowledge, and it is also common for senior sales staff to monitor the performance of junior sales staff as part of their training. For the sale of high-ticket items, there may be several parties on the buying side and several parties on the selling side.

•

To summarize, the praxis of selling goods and services has evolved in recent history to incorporate appropriate technology. The face-to-face direct sell, usually based in specialized retail premises, has been extended to include catalogue selling (based around postal mail) and call centres (based around the telephone). The growing success of the Internet as a medium for selling and buying products and services raises the question of how to employ the underlying capabilities of the Internet to improve the customer experience. The inventions described in this document make possible a range of novel and innovative systems for real-time multimedia interactions between customers and CSRs, with the goal of making on-line purchasing and service a productive and enjoyable experience.

[Use for intro wrt session routing case:

A call centre hosted on the Internet can have characteristics similar to a virtual community; there may be many long-lived multimedia communication sessions, each with many participants. Each session within the call centre is used for a different, but well-defined, purpose. Some sessions may host large multi-party discussion groups for a specific topic;

such sessions grow by additional participants simply joining the session, and the session has a life that may extend beyond the participation of any group or individual within the session. Other sessions may be created in order to host an intentional communication between two specific participants, rather like a user placing a telephone call to another user. In all cases, the participants of the communication session may join and leave the session independently of the life of the session itself.

With perhaps hundreds or thousands of communication sessions running simultaneously in a call centre, the choice of how to deal with a communication initiation request from a requesting party can become complex. The problem is that, unlike a conventional telephone call in which the caller addresses a specific individual using a single telephone number and a new session is unconditionally created for each call, a call initiation request may be best dealt with either by connecting the caller to an existing session containing other participants, or by creating a new session for the request.]

Summary of the Invention

According to the present invention, there is provided

Brief Description of the Drawings

A web interaction system embodying the invention will now be described, by way of non-limiting example, with reference to the accompanying diagrammatic drawings, in which:

- **Figure 1** is a diagram of a communication session abstraction of the web interaction system;
- **Figure 2** is a diagram of a session transport of the web interaction system;
- **Figure 3** is a diagram showing the functional layers of the system;
- **Figure 4** is a diagram of a "Shop with Friends" session scenario;
- **Figure 5** is a diagram of a "Page is Place" session scenario;
- **Figure 6** is a diagram showing in detail the sequence of operations involved in routing a customer to a session and participants to the session;

. **Figure 7** is a diagram of one possible physical configuration of the web interaction system in the situation where a call center is connected up to the system;

. **Figure 8** is a diagram illustrating the interactions between the components of the Figure 7 system upon a customer seeking to contact a call-center customer service CSR;

. **Figure 9** illustrates a customer service CSR (CSR) graphical user interface (GUI) desktop, the desktop having both a call-management GUI component for managing multiple calls, and customer-interaction GUI component for interacting with a specific customer;

. **Figure 10** shows a text-chat GUI sub-component of the Figure 9 customer-interaction desktop component when no call is selected;

. **Figure 11** shows the text-chat GUI sub-component during use for a selected call;

. **Figure 12** shows a page-push GUI sub-component of the Figure 9 customer-interaction desktop component;

. **Figure 13** is a diagram illustrating the functionality of the CSR desktop;

. **Figure 14** shows the Figure 9 call management desktop component showing a new call;

. **Figure 15** shows the call-management desktop component upon a second call being answered by the CSR;

. **Figure 16** shows the text-chat GUI sub-component when no media channel of this type is present for the selected call;

. **Figure 17** shows the call-management desktop component upon a call being dropped;

. **Figure 18** is a diagram of a customer GUI desktop based on applet technology;

. **Figure 19** is a diagram of a customer GUI desktop based on the use of a desktop proxy provided by the web collaboration service;

. **Figure 20** is a diagram of the functional components of a marketing BOT; and

. **Figure 21** is a diagram of the functional components of a content-monitoring BOT.

Best Mode of Carrying Out the Invention

General Structure of Communications Mechanism

Figures 1 to3 illustrate the basic functional concepts and elements of the web interaction system by which multiple parties can communicate with each other across the web (World Wide Web) using multiple media types.

As will be more fully described below, the basic high-level abstraction used by the system is that of a *communication session* 11 (Figure 1) to which one or more entities 12 A,B,C (participants) can be added or removed as required to implement a particular web interaction service 26 (Figure 3). The communication session abstraction 11 is modelled in the web interaction system by appropriate data structures and methods (for example, implemented as instances of a communication session object) for keeping track of a current session and its participants, and for effecting operations such as the adding and removal of participants. The functionality for creating, managing and implementing session instances is provided by a communications session manager 14 (shown in dashed lines in Figure3).

Associated with each communication session 11 is a *session transport* 15 (Figure 2) which is an abstraction of functionality for actually effecting data communication between endpoint systems 16A,B,C corresponding to the session participants 12A,B,C, this communication being through one or media channels 17a, 17b, etc. The session-transport functionality is typically embodied as a central session-transport manager 19 (Figure 3) and communication components associated with the end–points, the manager being responsible for creating and managing a respective session-transport object instance for each session transport where state data is held for that session transport.

Thus, the session manager 14 is concerned with the high level management and control of sessions whereas the session-transport functionality is concerned with the establishment and maintenance of the required media channels for the session transport that underlies each communication session.

The on-going operation of the session-transport functionality is largely independent of the corresponding communication session manager 14 except for when session creation / destruction results in the creation / destruction of an underlying session transport, and for

when the addition / removal of session participants results in the addition / removal of channels to/from the corresponding endpoint systems. The session manager 14 and the session-transport functionality are kept in step through "leg controllers" 20 (shown in Figure 3) which provide a way for a session 11 to pass to endpoint systems the information required to join a session transport, and to receive back related status information.

The communication session 11, session transport 15, media channels 17, and leg controllers 20 and considered in more detail below with particular reference to Figure 3. This Figure presents a layered view of how the various elements of the web interaction system inter-relate to each other, the specific scenario shown being that of two session entities connected to the same communication session. As is typical for a layered model, the functionality of the higher layers is implemented using that of lower layers.

Communication Session - A communication session 11is a representation of the *state* of a set of communicating entities. An entity (participant) 12 will in most cases be a person, although software automata or *Bots* can also be participating entities. By *communicating* is meant that entities are using one or media types to communicate, such as speech (audio), video, plain text, diagrams and illustrations, graphics, animations and 3D content, the kind of communications that are appropriate between human beings who want to share information. By *state* is meant the collective attributes of a specific session: the identities of the communicating entities, the media types in use, the pattern of distribution of information, global session properties (such as admission criteria), privileged members of the set, etc.

An instance of a communication session object is created and destroyed by communication-session factory 13 functionally embodied in the communication session manager 14.

Each communication session instance has associated functionality for carrying out the following operations in respect of the session it represents:

1. Maintain the set of session entities currently in the session, together with their connection state to the session.

2. Create a session transport instance using session-transport factory functionality 18. The session transport is preferably created in a lazy fashion, only when required. The identity (ID) of the session transport instance forms part of the state of the communication session.

3. Carry out a small number of session operations in respect of session entities, including the basic operations of adding and removing session entities to/from the communication session, and transferring session participation from one session entity to another.

Session creation/destruction and the carrying out of session operations are, of course, effected in compliance with the requirements of the overlying service 26.

If appropriately authorised in the context of a particular service, session entities can also perform simple autonomous operations that affect the communication session. Thus, a session entity can autonomously *Join* a communication session, if the connection details of the communication and session transports are known and the session entity has the appropriate privileges. Session entities can autonomously *Leave* the session. Both autonomous operations are arranged to cause the state of the communication session, including the set of session entities, to be updated and appropriate communication session events.

The Session Transport – As already indicated, each communication session requires an underlying session transport instance to communicate data (both real-time, and non-real-time) between session entities. A session transport can be implemented, for example, by an IP multicast group, a unicast IP conference, or, as in the specific embodiment to be described hereinafter, a tightly-coupled, group communication server.

The session-transport functionality (in particular, session-transport manager 19) maintains state data on the state of each session transport. The state of a session transport is the collective attributes of a session transport, namely the connection, authentication and

authorization parameters required to join in the data transport mechanism, the set of channels in the session transport, and the set of channel endpoints connected to the channels.

Session-transport factory functionality 18 of the session-transport manager 19 is responsible for creating / destroying instances of a session-transport object used to represent and permit the set up of each session transport (in particular, a session-transport object will hold the state data for the session transport it represents). Requests from a communication session object to the session transport factory functionality to create a session transport are parameterised with the characteristics of the required session transport. The session transport factory 18 uses this information to create an instance of the session transport object that satisfies those characteristics.

Media Channels - A session transport encompasses one or more media *channels* 17 where a *channel* is an instance of a multi-party communications path between *channel endpoints* 22. Typically, a channel is used to disseminate information of a given *media type*. Examples of media types are textual chat, voice chat, shared whiteboard, collaborative browsing, and real-time voice and video. A channel can be used for sending control information, e.g. media-channel signalling information, queue status information, or positional updates in a 3D virtual environment.

An instance of a channel is created within the context of a single session transport. A channel has a unique name within the session transport. A channel defines a communication path between the connected channel endpoints, that is orthogonal to all other channels associated with the session transport.

A *channel endpoint* 22 is an instance of an addressable communication source or destination. A channel endpoint has a unique name within the context of a channel. An instance of a channel endpoint is bound to a single, named channel.

A *media client* 24 (Figure 3) is an instance of a specialized function to send and receive data of a specific media type, using a channel endpoint 22 bound to the channel for the

media type. The media client 24 provides a specialized programmatic interface to the channel for communicating with other media clients of the same type. An application 25 wishing to communicate using a particular media type, does so by creating a corresponding media client 24 which then contacts the session-transport manager 19 to create a named channel appropriate for the specific media type, if one does not already exist. The media client 24 then tries to join the channel by creating a channel endpoint 22 and binding it to the channel.. The application 25 can then use the media client to send data to and receive data from other applications that are associated, via their respective media clients, with the other channel endpoints connected to the channel.

There are three modes for sending data on a channel 17 from a channel endpoint 22:

1. Data is sent to all channel endpoints connected to the channel, including the sender.
2. Data is sent to all channel endpoints connected to the channel, excluding the sender.
3. Data is sent to a specific channel endpoint in the channel, by specifying the channel endpoint address. No other channel endpoints connected to the channel receive that data.

An application 25 may join any number of sessions. For the or each session the application joins, it will seek to bind to a set of media channels previously specified to it by the communication session manager 14 in a media description.

The *media description* specifies the number and media type of channel instances in a session transport, and the connection details required to join the session transport (these connection details include the address and type of the session transport, and the authentication and authorization information required to join the session transport). The Session Description Protocol (SDP, RFC 2327) defined by the IETF is an example of a standard scheme that can be used to specify the media description. Other ways of specifying the media description are also possible, of course, - for example, the media description can simply identify the set of media clients to be instantiated by media applications, the address of the session transport, and the authentication information required to join the session transport.

Typically, the application 25 creates one media client for every media type supported by the session.

Leg Controllers Each communication session instance maintained by the communication session manager 14 needs to be able to communicate with the endpoint systems 16 that correspond to its session entities 12 in order to invite the endpoint systems to connect up with the session transport, and to monitor the states of connectivity of the endpoint systems and thereby the *connection state* of each session entity 12 to the communication session 11. A small number of connection states are defined that model the stages of a session entity joining and leaving a communication session. The states correspond to the notions of inviting a called entity to a session, alerting the called entity of the invitation, connecting to the session transport, being in a state of connection to the session transport, requesting a disconnect from the session transport, and being disconnected from the session transport.

The communication between a session instance and an endpoint system 16 corresponding to one of its session entities 12 is effected through a pair of leg controllers20, one in the communication session manager 14 and the other in the endpoint system 16. (The term *leg* is used because session diagrams, such as shown in Figure 1, have a number of legs attached to a session body, each leg representing a participant). The leg controllers 20 provide the signalling functionality and state machine functionality for inviting an endpoint system into a session transport and subsequently change and monitor the connection state of the entity.

More particularly, to invite an endpoint system to connect up with a session transport, the inviting entity creates a leg controller and contacts the endpoint system at a connection endpoint 21 of the system (the connection endpoint provides a unique address for the endpoint system and is used in a similar way as a telephone number in the PSTN). This results in a corresponding leg controller being created at the endpoint system (if not already present) after which the pair of leg controllers exchange "leg messages" that carry a variety of data, the most important of which is the media description of the session

transport which the endpoint system uses to set up media channels as already described. The state of connectivity of the endpoint system is also reported via the use of leg messages. As already indicated, the connection-state abstractions exchanged by the leg controllers represent high-level, logical participation in the session transport, and are independent of the communication mechanism used by the session transport. Typically, the connection states are:

For the inviting entity:

- an inviting state between when an invitation has been sent to an endpoint system to join a session and when a message is received back indicating whether the invitation has been accepted or rejected;
- a connecting state between the acceptance of the invitation by the endpoint system and when an indication is received that the endpoint system has connected to the session transport;
- an established state between when the endpoint system connects to the session transport and either when the endpoint system disconnects from the session transport or the inviting entity sends a message to the endpoint system requesting disconnection;
- a requesting disconnection state between when the inviting party requests disconnection by the endpoint system and when disconnection is reported back;
- a final state entered after disconnection has been reported from the endpoint system or the joining process fails before reaching the established state .

For the invited endpoint system:

- an initial state up until when an invitation to join a session is accepted. The significance of this state may vary depending on the role of the endpoint system. Thus, where the endpoint system is the initiator of a communication service request, the initial state effectively corresponds to the period between when the request is issued and when an invitation to join a session is received back since the invitation will generally be automatically accepted by the endpoint system. In contrast, where the endpoint system is one receiving an invitation which it did not instigate, the initial state lasts between when the invitation is received and when it is accepted by the human or automated operator of the system. In either case, the acceptance/rejection of the invitation is reported back in a leg message to the inviting system.

- a connecting state between the acceptance of the invitation by the endpoint system and when the endpoint system has connected to the session transport;
- an established state between when the endpoint system connects to the session transport and either when the endpoint system disconnects from the session transport or the inviting entity sends a message to the endpoint system requesting disconnection. This state may include sub-states; for example, where the endpoint system is a customer wishing to be connected to a CSR, then the established state may comprise a routing sub-state corresponding to the period when the customer is waiting for a CSR to join the session, and a conversing sub-state when the CSR has joined the session.
- a disconnecting state between when either the inviting party or endpoint system user requests disconnection from the session transport and when disconnection is achieved;
- a final state entered after disconnection from the session transport or upon the joining process failing before reaching the established state.

Both leg controller instances only exist for the duration of the participation of the invited endpoint system in the session transport.

The external interface to, and logical signalling used by, the leg controllers is independent of the mechanism used to transport the leg messages carrying the signalling information. Many different transport mechanisms for leg controller messages are possible. For example, Java JMS can be used or a system such as described in our co-pending patent application **XXXXX** that enables communication to the customer desktop through a firewall. Socket and SIP transports are other possible alternative implementation choices. By using the above mechanisms, a communication session instance can implement the session operations by translating operations into a sequence of operations on instances of leg controllers to change the connection state of the affected session entities. The operations can be described in terms of a simple leg algebra of addition and subtraction of legs into the communication session.

Layered Model – Having described in turn the basic concepts and elements of the web interaction system, it is worth considering from a broader perspective the layered model of the system shown in Figure 3. This layered view organises the system into four layers,

each representing a different logical view of the connectivity and communication between the various entities.

The four layers of the model are described below:

1. The *Service Layer* represents the service logic written by application developers to intelligently conference session entities into a communication session. A service 26 manipulates the connection state of a set of session entities 12 to a communication session 11, using only the elements of the Communication Session Layer. A service uses the communication session operations to invite entities to or disconnect them from the session, and uses the communication session events to monitor changes in connection state of the session entities. Many different services can be written, each using the underlying Communications Session Layer. At the service level, each endpoint system communicates with the service 26 via a service interface 29 that typically takes the form of service-specific web pages running in a browser application.
2. The *Communication Session Layer* offers a high-level view of the participation of session entities 12 in a communication session 11. Users of this layer deal only with very high-level abstractions of participation in a conference. The communication session 11, communication session factory 13, and session entity 12 are the principal elements of this layer. The communication session 11and session entity 12 uses the leg controller 20to invite the remote participant to join the session transport 15.
3. The *Connection Layer* represents the protocol, messages, events, state machine and operations used to invite a participant 12/16 to a session transport 15, and subsequently manipulate the connection state of the participant to the session transport. The leg controller 20 is the principal element in the connection layer. Connect and disconnect procedures offered by the connection layer are independent of the implementation mechanisms used for the transport layer. The connection layer elements use the operations, and consume the events offered in the transport layer. The events generated by the connection layer are used by the communication session layer to update the state of session entities 12 and the communication session 11.

4. The *Transport Layer* represents the elements involved the exchange of application data between session participants. The session transport 15 is instantiated by the communication session 11using the session transport factory 18. Channels 17 and channel endpoints 22 can be instantiated by any entity with sufficient privilege. For example, both entities in the connection layer and media clients can be authorised to instantiate channels.

The messages exchanged between functional entities in the connection layer can contain information from other layers, in addition to the specific information of the connection layer itself. Services in the service layer are able to pass arbitrary information as key-value pairs to session participants in the add and remove operations. The communication session layer uses the transport layer messages to send information to invited session participants describing the current state of the communication session abstractions, allowing session participants to reconstruct the current view of the communication session layer.

Service Scenarios

The above-described architecture of the web interaction system allows considerable flexibility in how a request from a user to communicate with one or more other participants is satisfied. How the request is handled depends on the characteristics of the service 26 to which the request is directed, it being the service that controls what session is involved in the communication (including whether this is a new or an existing session) and what other participants are invited to join the selected session.

A service is embodied as functionality for providing the desired service behaviour using the session resources available to it in the communication session manager 14 (session creation/destruction, the session operations described above for adding and subtracting session participants, and the feedback of session state information in event messages generated in response to session events).

Before describing the mechanisms used for routing a call initiator to a session and inviting other participants to that session, a number of service scenarios will be outlined to illustrate

the breadth of applications possible using the above-described communications architecture. Three general types of services will be described, namely:

- One-to-one customer/CSR interactions where a customer wishes to interact with a CSR at a call center;
- "Shop with friends" where several people wish to conduct coordinated browsing; and
- "Page is Place" where concurrent visitors to the same web page can communicate.

In outlining the service scenarios, the following terms are used:

- Text Chat. Each member of a session can type lines of text into a chat GUI at any time. These are sent to other session members in real time (which in practice may mean a delay of up to a few seconds) via a text media channel and displayed in a chat window, interleaved with the name of the person sending the text.

- Page Push. The page corresponding to a WWW URL is displayed in a reserved browser window of each session member. A media channel is used to convey URLs between participants to the session. In a "Follow-Me Tour", clicking on a hyperlink on the page in the Page Push window results in all session members following that link in synchrony. Page Push, and its variants, is a way for session members to share WWW content.

- Callback or Dialback. A Web session member can be called-back at their telephone number. This feature is common in telephony call centers, and a telephony call center will have dedicated hardware for terminating and routing incoming telephone calls to CSRs. This hardware will usually have the ability to originate calls, making it possible to set up a dialback call between a CSR and a customer. The capability of adding a dialback connection to an ongoing Web interaction is an example of hybridization between existing call centers, which are oriented around telephony, and the next generation of Internet Relationship Management centers which use Internet technology for communication with a customer.

- Deferred Callback. A customer is called back at a nominated time.

Customer/CSR 1:1 Interactions A number of different interactions are possible and each can be considered as constituting a service. These interactions include:

Online Help - A customer is browsing WWW pages belonging to an enterprise and wants to talk to a CSR in that enterprise. The page will have some kind of "Help" button hyperlink which the customer clicks on. The customer browser then progresses through a WWW dialog which makes it possible for the customer to identify themselves by submitting a small number of personal details (e.g. name, customer reference, email address etc). The customer browser then launches graphical user interfaces (GUI) for each of the media types used in a session. This will typically be page push, and text chat or voice chat. An available CSR is discovered, joins the session, and the customer and CSR can then begin to discuss the issue that caused the customer to request help. The session can be extended by inviting-in additional CSRs, and the "call" can be transferred to another CSR. The session can also be extended to include additional customers.

Online Help with Dialback - As in the "Online Help" service, but the customer provides a Public Switched Telephone Network (PSTN) number so that the CSR (in fact, the telephone callback hardware referred to above) can dial back to the customer, so creating a voice channel in addition to the other communication channel. The PSTN, rather than the internet, is normally used for voice traffic because it provides a higher quality channel for voice communication than Voice over IP at the current time.

Online Help with Deferred Dialback - As in "Online Help with Dialback", the difference being that the customer wants to talk to a CSR at some future specified time. This is useful eg when all CSRs are allocated, or the customer wants to reserve a callback at a convenient time. The customer goes through the initial dialog, provides personal details including a telephone number, and is provided with the URL of a page to return to at a later time. When the telephone callback occurs, the customer goes to the URL provided, and information about them previously stored in the WWW browser is used to identify the customer and callback. The session GUI is launched by the customer's WWW browser and connects to the correct collaboration session. The CSR is also invited into the same session.

Deferred Dialback - The customer uses the initial WWW dialog to select a telephone callback at a specified time. No web interaction session is created, and the Internet is not used as a communication technology.

Web Rendezvous - The customer is speaking with an CSR over the telephone and decides to do a web based communication for page push or chat. At this point the CSR presses the "Rendezvous" button which will generate a session identifier/password for the CSR to give to the customer. The customer goes to a URL that is the "Rendezvous page" and enters the session identifier/password. The web interaction session is setup as per a normal session except the call will go to the CSR that the customer is already talking with.

Shop with Friends (Figure 4)- This scenario assumes that two or more friends want to browse and make purchases online. They want to communicate with each other using text or audio chat, and see the same WWW pages in a "Follow Me" tour as described above. In **Error! Reference source not found.**, three friends 12J,K,L are depicted as viewing the same web page together, discussing its content (via an audio media channel), and they have invited a CSR 12X to answer some questions. From the customer perspective this scenario is identical to the "Online Help" scenario, with the exception that the session members are customers and not a mix of customers and CSRs.

The first friend (e.g. participant 12J) initiates the session using a WWW dialogue (to be more fully described hereinafter) to submit a small number of personal details, and is given a session identifier/password which needs to be communicated to all other friends by some other means (e.g. email, instant messaging). Each friend goes to a WWW page, types in the session identifier/password, and becomes a member of the session, and so is able to see the same WWW page as other session members (because of Page Push/Follow Me) and is able to engage in text or audio chat with other session members.

"Page as Place" (Figure 5) - The "Shop with Friends" scenario uses a fixed multi-party session, and a succession of WWW pages "flow throw" the session using follow-me page push. Session participants effectively wander around the WWW, the session maintaining

its coherence as it travels. An alternative is the "Page as Place" scenario, where a communication session is immutably associated with a specific web page. In this scenario, as customers move from page to page, they move from session to session. **Error! Reference source not found.** shows four women 12J,K,L,X looking at two different pages 30A and 30B, each being associated with a respective session 11A, 11B. The woman 12L is viewing page 30B and is in session 11B by herself. The two women 12J,K are both viewing page 30A and are therefore in the same communication session 11B and can communicate with each other via appropriate media channels; these women 12J,K have been joined by a third person 12X – a CSR who is monitoring activity on page 30A.

The advantage of this scenario is serendipity: it corresponds closely to what happens when a person wanders around a mall, meeting a different set of people in each shop. Wandering into a page showing lawnmowers, one can choose to see whether anyone else is also looking at lawnmowers, and engage them in conversation. One might see a CSR just "standing around" on the page, or one could listen in on what a CSR was telling another customer.

There is a great deal that can be done with this simple concept. Instead of thinking of a WWW site as a catalogue, it can be organized like a department store or a mall into a set of places – the perfume department, the coffee shop, ceramics, cooking and so on. Instead of a customer having to decide whether their query is important enough to justify contacting a CSR, they can see CSRs "standing around" when they move from page to page. It is not considered an imposition to approach an idle sales assistant in a shop even for the most trivial of queries, because we know that is what they are there for.

Session Routing

A description will next be given, with reference to Figure 6, of how a party at an endpoint system 16 initiates participation in a service-specific communication session. For simplicity, in the following, no distinction is made between endpoint system 16 and the participant party using the system.

In general, the initiating party16 will be requesting a specific service that is centred around a particular target subject such as a person, page, catalogue item, or any other concept that is meaningful both to the requesting party and the routing functionality that handle the requests. The selected service will involve communication with another participant or participants who are in some way associated with the target subject. The process of setting up the desired communication service for the requesting party is a two-step routing process:

- The first step is to select a communication session 11 for the initiating party 16 to join on the basis of factors such as the service selected, the target subject and the requesting party. The selected session will be either a pre-existing session or one created for the new call; in either case, there is an associated service instance 26 providing the service specific behaviour associated with the selected session. This first step is carried out by session initiation functionality 35 which creates a temporary initiation instance 37 for routing the requesting party to the appropriate session with the aid of session routing functionality 46.
- The second step (which is not always needed) is to extend the participants in the communication session 11 by selecting one or more other parties to invite to the session. 11. The second step is carried out by the service instance 26 associated with the selected session.

More particularly, when a requesting party 16 selects a specific service via a web interface in their browser 29, they are passed service-specific pages 34 from a web server 33 that provides a service front-end. These pages, and associated server-side scripts and servelets, are used to collect data about the requesting party, service options, target subject etc, which is passed to a service-specific initiation instance 37 that was created (by functionality 36) in response to the initial selection of the service concerned by the requesting party. This initiation instance 37 resides on the communication session manager 14 and its identity is returned to the server 33 so as to enable data collected from the party 16 to be correctly passed back to the instance 37 (by way of example, this identity could be held in an endpoint-system-specific session object on the server 33 with session cookies, including a unique requesting-entity identifier, being used to link received HTTP requests from system 16 with the session object). The initiation instance 37 is operative first to carry out a data

collection and collation task 38 to establish enough information to enable the right communication session to be selected, and (if appropriate) the right participants to be invited to join; this body of information is herein called the initiation context 40. Collecting the information necessary to complete the initiation context 40 is primarily done through the web pages 34 but may also involve lookups in a customer database 39 holding information about the requesting party, and potentially other relevant databases.

The information contained in the initiation context 40 will to some extent be service specific but will generally involve information grouped into the following data sets:

1. *Requesting party.* This data set is used to describe the characteristics of the requesting party. Examples are the name, e-mail address, physical address, country of origin, telephone number, gender, and profession. Other attributes could relate to preferences of the requesting party, such as an interest in sport and music. An important (though not necessarily essential) attribute of the requesting party is a unique user identifier, used by the system to identify the requesting party, and used as a key to database 39. Typically, this unique identifier is created the first time the requesting party visits the site, and identifies the requesting party for all subsequent interactions with the site.

2. *Communication endpoint system.* This data set is used to describe the communicating device 16 used by the requesting party, for example the media capabilities and name of the device.

3. *Target Subject.* This data set is used to describe the entity the requesting party wishes to establish a communication session with. For example, the abstract entity may simply be 'customer service representative', with additional attributes that describe a marketing campaign such as 'Vacuum cleaners'.

4. *Service data.* This data set is used to describe additional information required by the specific service associated with the selected called abstract entity.. For the example of the 'customer service representative' abstract entity, described by the 'Vacuum cleaners' campaign, the service data could correspond to a specific product range, feature set, or price range.

5. *Communication option.* This data set describes the preferred communication mechanism of the requesting party. The requesting party may wish to communicate by Internet or non-Internet channel, or some combination of the two. Non-Internet channels could be telephone, or fax. Internet channels represent a variety of multimedia data types such as text or voice chat, collaborative web browsing, Internet voice and video telephony. The communication session may be established either immediately or at a future time, specified by the requesting party. Often the communication option is service-specific. For example, the requesting party may want to communicate with other individuals with similar interests, with a customer service representative, or with an automated bot.

The sets of parameterised data, described above, are derived and collated from several sources:

- *Referrer URI.* The URI of the page that held the link to the first service-specific web page 34 for a particular service can provide valuable information as it represents a simplification of the browsing history of the requesting party.

- *Embedded in a Web page 34.* The Web page(s) 34 returned to requesting party 16 and used to request the establishment of a communication session, may contain arbitrary amounts of embedded data in the form of name-value pairs. The data may be statically embedded in the page, or dynamically generated by active server technology, such as Java Server Pages (JSP), as understood by those skilled in the art. For example, a page may contain parameters that describe high-level semantics of the page, such as the product sales campaign and a specific model number This data is extracted and passed back from the requesting-party browser 29 in the request for a communication session to the web server 33 and from there to the initiation instance 37. The data may be visible or invisible to the requesting party, the decision is made by the web site designer.

- *Input by the requesting party.* The requesting party may be presented with a form to input information about him/herself (name, e-mail address, postal address,

country, telephone number, age, gender, profession, and interests), to describe the communication option required (such as text chat, voice chat, page push, shared whiteboard, Internet voice, Internet video, and PSTN telephone call), or to select the target subject.

- *Persistent data in requesting party browser*. Information can be stored in the requesting-party browser 29 (i.e. in "cookies"), to describe or simply identify the requesting party, to maintain service state, or requesting party preferences.

- *Persistent information held centrally*. Requesting party information (such as name, address, *country*, telephone number, or service subscription options) can be persistently stored in the database server 39. The requesting party identifier is used as the key to the appropriate database entries. Also of interest can be the earlier browsing history of the requesting party through the pages of an enterprise website prior to making the service request. This history is called the Wake Repository.

Once the initiation context information has been collected, the initiation instance 37 executes a session routing task 45 with the aid of the session routing functionality 46. The session routing function consists of intelligent services to analyse the initiation context and decide whether to select an existing session in the session pool 31, or to create a new session (using session factory 13). An identifier of the selected session is returned to the task 45 by functionality 46, this identifier serving also to identify the session-associated service instance 26 which, in the case of a newly created session, is created and bound to the session 11 by the task 45 with the aid of functionality 47.

Upon the session and service instances being identified, the initiation instance 37 hands on subsequent processing of the service request to the service instance 25; in particular, the service instance is informed that a new participant, with associated initiation context 40, wishes to join the related session. The service instance now proceeds to invite the requesting party 16 to join the selected session 11 (task 50). The normal mechanism, previously described, for inviting an entity to join a session is to use the appropriate session operation to add the party to the session with the session then creating a leg

controller 20 through which it communicates with a corresponding leg controller 20 of the target entity to invite the latter to join the session. However, since the requesting entity 16 cannot be relied upon to have an instantiated leg controller 20 at a known address at this stage, either the passing of a joining invitation from the session leg controller must be delayed until a matching leg controller instance has been created in the requesting entity and its address communicated back to the session, or else an alternative invitation mechanism must be used. In one such alternative mechanism, after the selected session 11 has added the requesting entity and created a corresponding leg controller 20, the address of the latter and the session ID are passed to the requesting entity 16 (or, as will be seen, its proxy) via the communication path already established with the requesting entity through the service front end 27 (see chain-dashed arrow 51). The passing of this information effectively constitutes an invitation to the requesting entity 16 to join the session which it now does by creating a leg controller 20 and connecting with the corresponding leg controller previously established by the session 11. Leg controller functionality can be provided to the requesting entity 16 either by being passed to the entity 16 in the form of an applet from the service front end 27, or by having the latter act as a proxy for the requesting entity with the leg controller functionality being part of the proxy functionality.

The media description of the session transport associated with the selected session is now passed to the requesting entity 16 by the session (unless this was previously done when passing the session ID and leg controller address to the entity) and the requesting entity proceeds to establish appropriate media channels with the session transport instance (the latter having been previously created by the session instance 11).

Depending on the nature of the service, upon the requesting party joining the selected session, one or more further participants can be automatically invited into the session by the service instance 26 on the basis of the information contained in the initiation context 40 the current state of the selected session, and the nature of the service concerned. One typical example would be the invitation of a specialist CSR into a session in the case where the service was online contact with a CSR about a specialist topic. Figure 3 shows the service instance as carrying out task 52 to identify an appropriate additional participant to invite to the session, this task making use of a participant resource 54 (for example, a

contact center manager for identifying the next available CSR suitable to handle the subject of interest to the requesting party). Once an appropriate additional participant has been identified, that participant is invited into the session (task53). Frequently, the invited participant system is one, such as a CSR desktop system, that is pre-configured to form part of the web interaction system and is therefore provided with appropriate functionality - in particular, with persistent leg-controller instantiation functionality at a known connection address. In this case, the join invitation is issued through a corresponding leg controller 20 of the session to the known connection address of the participant system, thereby causing the instantiation of a leg controller in the participant system followed by an exchange of leg messages as already described above.

Note that although in the foregoing the selection of an additional participant was initiated by the join event of the requesting party 16, task 52 could equally well have been triggered immediately following session selection whereby the invitation into the session of the participant effectively occurs in parallel with the invitation to the requesting party.

Generally the participants to a session will not only be able to communicate with each other through the media channels established between them, but they will also be able to interact with the service instance 26 through web-page functionality served by the front-end server 33, the session ID being all that is needed to link any participant input to the appropriate service instance 26. This enables participants to invite further participants, such as a CSR into a session at any appropriate time, the tasks 52 and 53 being executed on the basis of existing information and any new information supplied with the invite request.

Specific Embodiment

Figure 7 shows one arrangement of equipment for implementing an embodiment of the above-described web interaction system in the case of a customer 60 connecting across the Internet 63 to an enterprise web server 64 and wishing to initiate web interaction services, including communication with a CSR via CSR desktop 74. As will be appreciated by persons skilled in the art, the names and quantities of servers hosting the web interaction services are shown for the purposes of illustration and clarity, and should not be read as determining a unique physical instantiation of the architecture. There are many physical

configurations that can satisfy the architecture, and the choice usually comes down to non-functional criteria such as performance, scalability, reliability, security etc.

The customer has a desktop system (for convenience embraced by reference 60) which is connected to a LAN 61 located within a customer firewall 62. It is common in many organisations to use a firewall to isolate the private internal network from the public Internet for security reasons. The customer 60 could be a domestic consumer connected to Internet 63 via an Internet Service Provider (ISP) or it could be an employee of an organisation with a high level of internal security, such as a bank or a hospital. The customer 60 is connected through the customer premises firewall 62 to the Internet using the standard Internet TCP/IP protocol, and the customer has World Wide Web access using the standard Hypertext Transfer Protocol (HTTP).

The enterprise web server 64 is connected to enterprise LAN 66A which connects to Internet 63. Web server 64 resides within the so-called "demilitarized zone 65" of the enterprise, this being a ring-fenced LAN which includes equipment that is controlled by the enterprise but is accessible to the outside world as well as to equipment on the secure part of the enterprise network (LAN 66B) that exists behind a firewall.

In browsing the pages served by the enterprise server 64, the customer decides to request an offered web interaction service and indicates this by an appropriate selection action that results in a corresponding HTTP request message being sent to the web interaction system, the front end of which (functionality 27 of Figures 3 and 6) is embodied in a session mediation server 67 that also lies within the DMZ 65. Other equipment components of the web interaction system include a Communication Session Manager (CSM) server 69 providing the functionality of the communication session manager 14 of Figures 3 and 6, and a Thin Client Group Communications (TCGC) server 70 providing the session transport functionality 19 of Figure 3 ("Thin Client" is used to refer to the approach whereby most of the functionality resides in a server). The servers 69 and 70 both reside behind the DMZ 65. More details of the implementation components of the Figure 7 web interaction system are given below.

Thin Client Group Communication Server 70 - This component can be implemented, for example, using Sun's Java Shared Data Toolkit (JSDT). One or more TCGC servers 70 can be instantiated, to provide scalability. The TCGC server exports a simple interface to the session transport factory 18 (Figure 3) to allow other functional entities to create and destroy new session transports. The session transport is created with a set of authorisation parameters which are passed across the factory interface. In the current embodiment, session instances 11 residing on the CSM server 69 are the only authorised clients of the session transport factory interface.

The session transports 15 provided by the TCGC server 70 are centrally managed. The TCGC server is responsible for authenticating entities that attempt to perform operations on the session transport 15, channels 17 and channel endpoints 22. In particular, session transport creation and destruction, creating or obtaining a reference to a channel, and binding a channel endpoint to a channel, are all operations that need to be authenticated by the TCGC server 70. The TCGC server has full knowledge of the set of channels 17 created in a session transport, and the set of channel endpoints 22 bound to the channels.

Communication between channel endpoints 22 is implemented using a unicast transport. The originating channel endpoint 22 sends data to the TCGC server 70, and it is the server that forwards the data on the unicast transport to each of the destination channel endpoints 22 for the channel concerned.

Communication Session Manager Server 69 - The CSM server 69 provides the platform on which the functionality is deployed to create and manage communication sessions and their associated service instances. The realisation of the initiation instances 37, service instances 26, communication sessions 11, communication session factory 13, and server-side leg controllers 20 are part of the CSM. The CSM holds the definitive view of the communication session state.

In the current embodiment, the CSM offers a Java RMI remote interface to allow servlets running on the SMS (see below) to create and communicate with instances of the services deployed on the CSM.

Session Mediation Server 67 - The Session Mediation Server (SMS) 67 is a web server that hosts the set of service-specific pages that present the GUI to allow a requesting party to request connection to a service-specific communication session concerning a target subject. The GUI allows the requesting party to select a service, enter personal information, select a communication option, and describe the entity to communicate with. In the current embodiment, the server is, for example, an Apache Web Server running the Jserv Java Servlet environment. Java Servlets running on the server are used to parse WWW forms and validate customer inputs, and these servlets use the above-mentioned RMI interface to the initiation and service instances running on the CSM to satisfy the communication requests of the requesting party.

Other Components – The web interaction system also includes other components connected to the enterprise LAN 66B, these being a customer database server 39, additional information servers 71, a contact center manager server 72 for providing the CSM 69 with information regarding the availability of CSRs in CSR pool 73 (the CSRs and their desktop systems being indicated by references 74 in Figure 7)

Interaction Scenario

Figure 8 depicts the communication protocols used between the above-described system components. These protocols include the following, all are layered on top of the standard TCP/IP protocol:

- Hypertext Transfer Protocol (RFC1945). This is the protocol used by WWW browsers to access WWW servers for the purpose of providing the WWW service. This protocol is the lowest-common-denominator means for customers to interact with the web interaction system, and it can be transported securely across corporate firewalls. The customer desktop system 60 uses this protocol to communicate with the enterprise web server 64 and the SMS server 67.
- Java Remote Method Invocation (RMI) (see javasoft.sun.com) is a protocol used to invoke methods on remote objects in distributed systems. It is here used between the SMS server 67 and the CSM server 69 for invoking service and session initiation operations provided by the CSM.
- Java Message Service (JMS) (see javasoft.sun.com) is a public specification for inter-computer messages which has been widely implemented to disguise various

proprietary protocols, and is a convenient way to specify and implement interactions between the CSM 69 and TCGC server 70, as well as between the CSM and back-end servers such as databases 39 and 71, call centre manager 72, and CSR desktop applications 74.

The following scenario illustrates the use of these protocols during the initiation of a basic and familiar service where a customer 60 browsing the WWW wants to contact a CSR 74 to ask questions. The service takes information about the customer, creates a communication session, invites the customer into the session, locates an available CSR, invites the CSR into the session, and once both parties are in the session, they can begin to interact using various media. The general interactions involved are referenced [1] to [8] in Figure 8 and are as follows:

[1] The customer 60 is browsing the enterprise WWW site on server 64 (using HTTP protocol) and wants to talk to a CSR 74 about some issue. The customer finds a "help" button on the WWW page currently being viewed and clicks on it. This button is a hyperlink to a WWW page on the Session Mediation Server 67.

[2] The customer goes through various hypertext/WWW dialogues on the Session Mediation Server (SMS) 67. This involves selecting various communication options, and the customer supplies a small amount of personal information. The servlet running on the server 67 also extracts information about the customer in the form of "cookies" and other information from the HTTP header request, as already described.

[3] The SMS 67 condenses all the information about the customer in the form of a Java object and communicates it to a service-specific initiation instance on the CSM using the Java RMI protocol (as an alternative to passing all the required information in one go, this can be done progressively as the SMS collects information). There may be additional communication (using the JMS protocol) between the SMS/CSM and external database 39 containing customer information, so that the information presented to the customer can be personalised according to the past history of interactions.

[4] The initiation instance on CSM 69 causes the creation of an empty communication session 11, with associated service instance 26, for the customer. The session

instance 26 communicates (using the JMS protocol) with the TCGC server 70 to create a session transport 15.

[5] Information about the session and the session leg controller corresponding to the customer are then returned via the SMS to the customer along with the customer interface to the session (including leg controller functionality and the other functionality to be a member of a session) as part of an HTTP response (i.e. a WWW page containing active content such as Java or Javascript). In this way the customer loads a WWW page which contains information about the selected session transport.

[6] The customer system 60 joins the session 11 by using the leg controller functionality passed to it to contact the corresponding leg controller on the CSM server 69. Because of the difficulties involved in traversing the customer's firewall, the leg messages passed between the leg controllers actually use the same transport mechanism as employed for the media channels, namely a firewall traversing protocol, such as that described in our co-pending patent application XXXX, depicted by a chain-dashed line in Figure 8. Indeed, the leg messages can conveniently be passed across a channel established for this purpose between the customer end system and the session transport 15. To this end, the media description is passed to the customer system along with the session identity via the SMS whereby the customer system can establish communication with the session transport 15 using a firewall-crossing protocol and set up a channel to pass the leg messages as well as the required media channels for each of the media types in the session.

[7] The service instance 26 on CSM 69 interacts with the contact center manager 72 (using JMS) to select an available CSR.

[8] The service instance 26 invites the CSR desktop system 74 to join the session using the JMS protocol. This invitation contains customer information.

[9] If the CSR accepts, the CSR desktop system 74 joins the session transport suing the same protocol as the customer desktop system.

At this point, both the customer and CSR can exchange information using media channels created as elements of the session transport 15. The CSM 69 monitors the status of the end system participation in the session transport via the leg controllers, and when either customer or CSR leave, it tears down both the communication and session transport.

Although in step [5] above, the SMS is described as passing the customer system the functionality needed to join and be a member of a session, as a collection of Java servlets and Java packages in a WWW page, this functionality could be retained at the SMS with the latter acting as a proxy for the customer system and only serving to that system WWW pages that reflect the service/session state as known to the proxy.

This physical infrastructure is capable of supporting many service scenarios such as those described above.

Endpoint System Desktops

Examples will now be given of GUI desktops suitable for a CSR endpoint system 74 and a customer system 60, these desktops being the visible expression of the functionality described above. In particular, the desktops provide interfaces for the media channels associated with a communication session as well as web interfaces to the web interaction service 26 concerned; additionally, the CSR desktop provides tools for managing multiple "calls" (communication sessions).

CSR Desktop (Figures 9-17)

The CSR Desktop 80 is the CSR's sole point of interaction with web channel calls but may be used in conjunction with other channels, e.g. telephony, to offer richer service, e.g. voice and page push. A CSR is associated with one or more *campaigns* which are a way of breaking down a customer service problem space into smaller logical areas; typically, a specific campaign will constitute the target subject matter of a customer service request.

In telephony, interaction between a customer and CSR is swift and if the CSR wishes to split their attention between two or more calls, they must put one of the calls on hold, thus breaking the appearance of dedicated service. With the web channel, interaction can be less intense, e.g. if the customer is not familiar with a keyboard, and there is opportunity for an CSR to multi-task between a number of calls. To support the illusion from the customer's perspective that the CSR is giving them dedicated service requires a CSR desktop GUI component that enables the CSR to manage the information and media

associated with each call quickly and effectively. Figure 9 shows an example CSR desktop GUI 80 with a call-management component 82 that can be used by the CSR to receive incoming calls and manage calls that they are already dealing with. Each call (a session that the CSR is invited to, is currently involved in, or has recently left) appears as a row in a table containing relevant information such as customer name, customer ID, the campaign this call belongs to and the status of the call. To interface with a particular call, the CSR selects the row containing the call details (and possibly is also required to press an appropriate button).

The call management component 82 includes a set of high-level control buttons 81 for choosing actions such as accepting/rejecting an invitation to join a session, disconnecting from a call, transferring the call to another CSR and conferencing in another CSR

The CSR desktop 80 also comprises a customer-interaction component 83 with respective GUI sub-components for each of the media types that the desktop 80 is intended to handle (albeit that not all media types are present in all calls). These GUI sub-components are generically called media-type windows below with the sub-components associated with a specific media type being referred as that media-type window. In the Figure 9 example, a text-chat window 85 is shown together with a page-push window 86 and a browser window 87 (the latter two windows both being used for a page-push channel, the browser window showing the page at the last-pushed URL as displayed in the page-push window 86). Upon a call being selected in the call-management window 82, the media clients established for the call are linked to the corresponding media-type window.

Example media-type windows are shown in Figures 10 –12. More particularly, Figure 10 shows a text-chat window 85 in the case where no call has been selected, whereas Figure 11 shows the same window 85 when a call, with previous chat dialogue, has been selected. Figure 12 shows the page-push window 86.

Figure 13 depicts the general arrangement of functionality supporting the CSR desktop 80. A central control block 90 comprises leg controller functionality 91, and a session manager 92 for managing the sessions in which the CSR desktop system is involved. When the leg

controller functionality 91 receives a leg message inviting the CSR desktop system to join a session, it creates a corresponding leg controller in the alerting state and causes the session manager to create a new line in the call management window 82. This line is emphasised in some manner (e.g. shown in red) to alert the CSR to the invitation (see sole line in the call-management window depicted in Figure 14). Upon the CSR accepting the call by selecting the line and clicking an accept button, an appropriate icon is added to the status field of the call line (see Figure 15) and the session manager 92 instantiates media clients 24 for the media types indicated in the media description of the session. These media clients then set up corresponding media channels to the session transport as already described. The state of the leg controller passes to 'connecting' during channel set up and then to 'established' once the channels have been set up.

For whichever session is currently selected (the currently selected session is highlighted in blue with only one call being in a selected state at one time), the session manager causes the media-type windows 85 – 88 to display the output of the corresponding media channels of the selected session. Each user interface function is disabled or enabled depending upon the status of the currently selected session to ensure that the user is only presented with the options relevant at any given moment; in particular, if a particular media type is not required for a selected session this is clearly indicated and the relevant window controls disabled (see Figure 16 that shows the text chat window 85 when not being used for a currently-selected session).

While dealing with one call, some new content may appear on the media channels associated with one of the other calls being handled by the CSR. When this occurs, an icon representing the media type of the new content is displayed in the call table's media column. This simple mechanism enables a CSR to concentrate on interacting in one session in the safe knowledge that they are not missing input from another customer. As soon as the media icon(s) appear, they can select that call and check what has happened, making a response as necessary (when a call is selected, all media icons are cleared from the call's display). Using this process, a skilled CSR should be able to handle a number of calls simultaneously.

When the customer or CSR drops the call, the call details do not disappear immediately. Instead, the call table entry remains (see second row in the call management window 82 shown in Figure 17) and, if the call is selected, the media-type windows can be used by the CSR to review the content generated when interacting with the customer. When the CSR is happy that they have captured all the information they need, the call can be removed from the call table and the media content is lost forever (unless archived).

Scripts - CSRs often have to deal with the same types of enquiries over and over again. To extract the most relevant information as efficiently and as quickly as possible, a CSR will often ask a series of predefined questions. These are commonly placed into written scripts which are either followed from top to bottom, or sampled as necessary. When using telephony this is sufficient. However, when presented with a medium of interaction such as text-based chat, the CSR is required to type repetitive phrases and questions in addition to the usual conversational pleasantries. The menu bar 95 in the Figure 11 text-chat window 85 shows two scripts that are represented by pull-down menus 96, 97. The first menu 96 (labelled "General Phrases") contains commonly used phrases such as "Hi, how can I help you?", and so on. The second menu 97 (labelled "Hoovers") contains questions that are specific for the campaign related to the call, e.g. "Hoovers". As a CSR flicks from one call to another, the correct campaign script is automatically displayed in the text-chat GUI.

Similar pull-down menus 98, 99 are also available in the page push window 86 (see Figure 12) and contain URLs in much the same way as browser provide a bookmark facility. These URLs are again ordered for general campaign-independent use (menu 98) and campaign-specific relevance (menu 99). Selecting one of these URLs will push the page to the session.

A CSR in using the script-enabled text chat and page push windows to show a customer around a product, for example, would say (type in the text chat window 85) something or choose an entry from a script menu, and then select the appropriate URL from one of the menus 98, 99 in the page-push window 86. It is also possible to arrange for the selection of

particular text-chat script entries to automatically cause a corresponding URL to be sent to the page push media client of both the CSR and customer.

The above-described use of predefined selections, as well as their linking together across media types, can be applied to all media clients, whereby a multimedia presentation is effectively scored by the selection of script items.

Languages - In addition to being assigned to one or more campaigns, a CSR may also be multi-lingual and capable of dealing with enquiries in multiple languages. The country and language (together making a "*locale*") of the customer is passed to the CSR desktop as part of the call presentation and enables the media-type windows to change their output and input to display content appropriate to different locale and create content that will appear correctly on the customer's desktop, respectively. In the case of the text chat window, this results in the menu bar 95 displaying script names and their contents in the appropriate language. If the scripts in the appropriate language have not been previously loaded scripts then they are loaded upon demand. This allows the allocation of multi-lingual CSRs to change at run-time as well.

Call Operations - Control buttons 81 (see Figure 17) make available the following basic call-handling functions to a CSR :

Answer – (button 81A) Accepts the selected call and connect to the session.

Decline - (button 81B) Refuse the selected call such that another CSR will be selected to take it.

Drop - (button 81C) Used when CSR has finished dealing with the customer.

Transfer to CSR - (button 81D) Transfer the call to another specified CSR. If the receiving CSR accepts the call the desktop waits until all the receiving desktop's media clients are connected before automatically disconnecting their own for that call.

Conference to CSR - (button 81E) Sometimes, a CSR may wish to conference in another CSR, for example one with more knowledge on a certain matter. If the receiving CSR accepts the call, then the session is extended to include that CSR's

media clients. As long as there is one CSR dealing with the call, the communication session will remain open.

Clicking a control button 81 results in the session manager 92 initiating the appropriate action, including the sending of leg messages to the CSM server 69 to notify the session instance 11 of a change of communication state of the CSR desktop system (for Answer, Decline, Drop actions), and the sending of messages to the service instance 26 to add/remove participants (Transfer/Conference actions).

Customer Desktop

The customer endpoint system 60 provides a customer desktop interface the customer with the service being provided by the web interaction system. The following description of the customer desktop relates to the specific case of a desktop configured for customer-CSR interaction and it will be appreciated that for other service scenarios the precise details of the desktop will vary to suit the service.

The customer desktop typically proceeds through the following connection states that are reported to the communication session manager in leg messages:

- an initial state entered on launching of the desktop and ending when an invitation is received and accepted from a session;
- a connecting state between acceptance of the invitation and when the appropriate media clients have been set up and media channels established to the session transport;
- a routing state (described above as a sub-state of the established state) lasting whilst a CSR is being found and invited into the session;
- a conversing state (also described above as a sub-state of the established state) whilst the customer and CSR are both connected to the session;
- a disconnecting state, and
- a final state.

The CSR may transfer the call to another CSR. If this happens the customer desktop is informed and the customer informed so that they do no anticipate any further interaction until the call has been successfully transferred.

As with the CSR desktop, the customer desktop preserves the relevant media content, e.g. the chat transcript, after a call has terminated so the customer can review the information they gathered after the CSR has disconnected.

The Customer Desktop is composed of two layers: one providing service-level logic and functionality (the service interface 29 of Figure 3), the other implementing the media clients with their graphical user interface. Two functionally similar embodiments of the customer desktop are described below, namely:

- an applet-based customer desktop ("ACD") implemented, for example, using Java applet technology ; the ACD requires downloading to the customer endpoint system and carries out media channel processing on the customer desktops; and
- a proxy-using customer desktop (the "Lite" customer desktop, LCD because no download or extensive processing is required on the client side, the processing being done in a proxy provided in the SMS 67).

Applet Customer Desktop (ACD)

The general functional structure of the ACD is shown in Figure 18 and, as can be seen comprises a web browser 100 in which standard HTML pages (with embedded JavaScript) provided by the SMS 67 provide the service interface 29, whilst the browser's Java Virtual Machine 102 runs a downloaded applet 103 to offer a richer interface to the web interaction system.

The ACD is launched when the SMS 67 serves an HTML page containing the applet to be opened in the customer's web browser (this is normally done after initial information has been collected, using HTML forms, by the service front end running on the SMS 67). The parameters for the HTML applet tag specify session information needed to initialise the desktop and connect to the session leg controller (on CSM 69) and the session transport TCGC server 70, these parameters typically including:

- Customer identifier.
- Nickname (from the form filled in by the customer prior to desktop launch).
- Language the customer requested to receive service in.

- Address of the communication session associated with the call.
- The response to be given when challenged by the TCGC upon joining the session.
- The media description for the given session.

The first action performed by the applet 103 is to interpret the media description and create a media client 24 for every media type contained therein. All the media clients 24 are connected, via transport layer106,to the session transport at the given address and use the given response when challenged. At the same time, a single CSM client 105 is created and a connection with the CSM established, via the session transport 15, for the passing of leg messages to communicate connection state information to the session instance 11. As already indicated, the transport layer 106 preferably implements a firewall-crossing protocol.

The ACD's user interface is initialised so that input and output are suitable for the specified language and media GUIs created for the respective media clients. In this way, the user interface only contains those GUI elements that are strictly necessary for a given call.

The ACD uses an interface defined in Javascript to perform operations on and reflect certain events into the HTML document containing the applet. These events can be used for synchronising web content with the desktop. Just as the applet uses Javascript to access the HTML document, so Javascript can be used by the document to access the external interface of the ACD, e.g. get the current state of the desktop. For example, where a page-push media client is instantiated, this can cause the opening of a browser window to be used to displaying the pages pushed by the page push media client. Upon receipt of a new URL from the page push channel, the page push media client invokes a Javascript method to display the URL in the page push window.

Lite Customer Desktop (LCD)

The general arrangement of the LCD and associated proxy is shown in Figure 19. The LCD uses HTML and Javascript in the web browser 100 and locates the media-client and leg-controller functionality 24,105 in the SMS 67 (again, this functionality can be implemented using Java code 108 running in JVM 109). When the LCD is launched (done

by the SMS 67 serving the appropriate HTML pages to the customer system 60), a desktop proxy process 109 is created in the SMS that connects to the TCGC 70 to set up the required media channels and interacts with the session leg controller on the CSM 69. The LCD forwards any user input, e.g. chat message, page to push, etc., to the proxy 109 and polls it using an HTTP request for client updates, e.g. change in desktop state, new chat messages, page to display, etc.

Subject: Smartweb patent applications
Date: Tuesday, October 17, 2000 6:39 AM
From: Squibbs, Robert <Robert_Squibbs@hplb.hpl.hp.com>
To: "LOW, COLIN (HP-UK,unix1)" <colin_low@hpl.hp.com>, "Vickers, Paul" <paul.vickers@hp.com>, "HAWKES, RYCHARDE (HP-UK,unix1)" <rycharde_hawkes@hpl.hp.com>, "Wilcock, Lawrence" <lawrence_wilcock@hp.com>
Conversation: Smartweb patent applications

Gentlemen,

I am finally getting the 12 Smartweb patent applications filed today (hopefully). I am filing all of them in the UK Patent Office with novelty search requests. I am also filing the first case 30004631 in the European Patent Office to see if their searching differs radically from the UK Patent
Office.

All the patent specifications are available at:
http://w3.hpl.hp.com/legal/innov/smartweb/index.htm
in Word and PowerPoint. As you will see, the specific description and drawings are the same for each case (the ull texts thus only differ in their
claims and in their introductory portions).

You will also find two summary documents: one containing all the abstracts,
and the other all the main claims (claims 1).

Could you please review at the least the claims (and, most importantly, claim 1) of each case. All the claims 1 are in the summary document whilst the full claim sets are in the full texts (at the end, as usual). Please let me have your comments regarding scope of the claims. Whilst I have gone
ahead with filing the cases, if any of the main claims need serious revision, I shall file a supplementary application.

Best regards

Robert

EXHIBIT G

Claim Element	Exhibit
Claim 19	
An automaton for providing media content to media channels of a network communication session, the automaton comprising:	
a manager system configured to:	Exhibit A, pages 27-35; Exhibit B, pages 1 and 2.
join the automaton to an existing network communication session between an endpoint entity and a contact center responsive to receipt of an invitation to join the existing network communication session, and	Exhibit A, page 34, section 7.8.2, 7.8.3, pages 12-16 (see e.g., section 6.2), pages 27-28 (see, e.g., section 7.4); Exhibit B, pages 1 and 2.
receive: (a) context data about the existing network communication session and (b) channel information about one or more media channels of the existing network communication session, wherein the channel information includes media type carried by the media channels and channel connection details;	Exhibit A, pages 31-35, pages 32-34, section 7.7 and pages 26-27, section 7.2; Exhibit B, pages 1 and 2, pages 10-11 (see e.g., section 7.2), and page 12, section 8.
a transport system configured to establish, based on the received channel information, one or more media channel connections from the automaton to a session transport mechanism associated with the existing network communication session;	Exhibit A, pages 25-35, pages 26-27, section 7.2 and page 27, paragraphs 5-6; Exhibit B, page 12.
a media content handler configured to deliver media content of a particular media type to the established one or more media channel connections based on the received channel information; and	Exhibit A, pages 25-35, page 26, section 7.2, pages 27-32 (see e.g., section 7.6.1 "GetSessionAttributes").
a delivery controller configured to control the selection and delivery of media content by the media content handler responsive to the received context data.	Exhibit A, pages 25-35, page 16, 3rd full paragraph, page 26, section 7.2, pages 27-28, section 7.4.

Claim 36	Exhibit
A method of providing media content to media channels of a network communication session, the method comprising:	Exhibit A, pages 25-35; Exhibit B, pages 1 and 2.
establishing a media channel connection from an automaton to a session transport mechanism associated with an existing network communication session between an endpoint entity and a contact center responsive to receipt of an invitation to join the existing network communication session and receipt of channel information about one or more media channels of the existing network communication session, the channel information including the media type carried by the one or more media channels and channel connection details; and	Exhibit A, page 34, section 7.8.2, 7.8.3, pages 12-16 (see e.g., section 6.2), pages 27-28 (see, e.g., section 7.4), pages 26-27, section 7.2, page 27, paragraphs 5-6, and pages 27-32 (see e.g., section 7.6.1 "GetSessionAttributes"); Exhibit B, pages 1 and 2.
providing an appropriate media content from the automaton to a corresponding media channel established by said establishing step responsive to receipt of context data about the existing network communication session and based on the channel information.	Exhibit A, pages 31-35, pages 32-34, section 7.7 and pages 26-27, section 7.2, page 16, 3rd full paragraph, and pages 27-28, section 7.4; Exhibit B, pages 1 and 2.; Exhibit B pages 10-11 (see e.g., section 7.2), page 12, section 8.

\

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of	:	
	:	
HAWKES, RYCHARDE JEFFERY *et al.*	:	Confirmation No. 1484
	:	
U.S. Patent Application No. 09/977,501	:	Group Art Unit: 2143
	:	
Filed: October 16, 2001	:	Examiner: Jude Jean Gilles

For: CONTENT PROVIDER ENTITY FOR COMMUNICATION SESSION

Declaration Under 37 C.F.R. §1.131

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I, Lawrence Wilcock, based on information and belief, do hereby declare that:

1. I am an inventor of the invention described in U.S. Patent application serial number 09/977,501 (the '501 application) and filed on October 16, 2001 and described in U.K. Patent Application No. 0025454.0 (the U.K. Application) filed on October 17, 2000 and from which the '501 application claims priority.

2. At the time of the invention, I was employed by Hewlett-Packard Company (HP). I am currently employed by HP.

3. I conceived at least a portion of the invention claimed in claims 19 and 36 of the above-identified patent application prior to February 29, 2000 in the United Kingdom and diligently worked the invention until a constructive reduction to practice in the United Kingdom as demonstrated by the Attached Exhibits and the above-noted '501 application.

4. Prior to February 29, 2000, I contributed to the generation of a disclosure of the invention in the form of a document entitled, "A Web Interaction System." Attached as Exhibit A is a copy of the disclosure document. Exhibit A describes a Web Interaction system created for CRM, E-commerce, and customer interaction applications and services.

5. Prior to February 29, 2000, I contributed to the generation of an additional

disclosure of the invention in the form of a document outlining potential patent claims entitled, "Appendix 1 - Potential Patent Claims." Attached as Exhibit B is a copy of the document.

6. Prior to February 29, 2000, one of my co-inventors, Mr. Colin Low, generated and transmitted an email including exhibits A and B to Mr. Robert Squibbs of HP for use in preparing the '501 application. Attached as Exhibit C is a copy of the email message.

7. During the period between just prior to February 29, 2000 until October 17, 2000, I worked with Mr. Squibbs and provided information regarding the invention assisting Mr. Squibbs to draft the U.K. Application. As evidence of my interaction over this time period, attached Exhibit D is a copy of another detailed document describing information related to the Exhibit B document to which I contributed. Attached Exhibit E is a screenshot of the Microsoft Word file properties dialog displaying the date of creation of the file corresponding to Exhibit D. As further evidence of my interaction over this time period, attached Exhibit F is a copy of a version of the specification and drawings which I received from Mr. Squibbs during the afore-mentioned time period. During the afore-mentioned time period, I reviewed the application in order to provide comments and correction to Mr. Squibbs. During the afore-mentioned time period, I received an email message from Mr. Squibbs requesting me to review a revised version of the claims of the application prior to filing of the application. Attached as Exhibit G is a copy of the email message I received from Mr. Squibbs requesting the revised review. Exhibits D, F, and G were transmitted and/or received during the period between just prior to February 29, 2000 until October 17, 2000.

8. Attached as Exhibit H is a table comparing the elements of claims 19 and 36 with the corresponding conception identified in Exhibits A and B.

9. Exhibits A-G, which relate to the aforementioned conception of the claimed invention followed by diligence until a constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Actual dates of Exhibits A-B have been removed, but are prior to February 29, 2000. Exhibits D, F, and G, which relate to the aforementioned diligence until constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Further, company proprietary information has been removed from all

exhibits.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

L. Wilcock Date: 9th October 2006

Lawrence Wilcock

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400
Telephone: 703-684-1111
Facsimile: 970-898-0640

RAN/dll

Docket No.: 30004635-2 (1509-226) **PATENT**

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of	:	
HAWKES, RYCHARDE JEFFERY *et al.*	:	Confirmation No. 1484
U.S. Patent Application No. 09/977,501	:	Group Art Unit: 2143
Filed: October 16, 2001	:	Examiner: Jude Jean Gilles

For: CONTENT PROVIDER ENTITY FOR COMMUNICATION SESSION

Declaration Under 37 C.F.R. §1.131

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I, Rycharde Jeffery Hawkes, based on information and belief, do hereby declare that:

1. I am an inventor of the invention described in U.S. Patent application serial number 09/977,501 (the '501 application) and filed on October 16, 2001 and described in U.K. Patent Application No. 0025454.0 (the U.K. Application) filed on October 17, 2000 and from which the '501 application claims priority.

2. At the time of the invention, I was employed by Hewlett-Packard Company (HP). I am currently employed by HP.

3. I conceived at least a portion of the invention claimed in claims 19 and 36 of the above-identified patent application prior to February 29, 2000 in the United Kingdom and diligently worked the invention until a constructive reduction to practice in the United Kingdom as demonstrated by the Attached Exhibits and the above-noted '501 application.

4. Prior to February 29, 2000, I contributed to the generation of a disclosure of the invention in the form of a document entitled, "A Web Interaction System." Attached as Exhibit A is a copy of the disclosure document. Exhibit A describes a Web Interaction system created for CRM, E-commerce, and customer interaction applications and services.

5. Prior to February 29, 2000, I contributed to the generation of an additional

disclosure of the invention in the form of a document outlining potential patent claims entitled, "Appendix 1 - Potential Patent Claims." Attached as Exhibit B is a copy of the document.

6. Prior to February 29, 2000, one of my co-inventors, Mr. Colin Low, generated and transmitted an email including exhibits A and B to Mr. Robert Squibbs of HP for use in preparing the '501 application. Attached as Exhibit C is a copy of the email message.

7. During the period between just prior to February 29, 2000 until October 17, 2000, I worked with Mr. Squibbs and provided information regarding the invention assisting Mr. Squibbs to draft the U.K. Application. As evidence of my interaction over this time period, attached Exhibit D is a copy of another detailed document describing information related to the Exhibit B document to which I contributed. Attached Exhibit E is a screenshot of the Microsoft Word file properties dialog displaying the date of creation of the file corresponding to Exhibit D. As further evidence of my interaction over this time period, attached Exhibit F is a copy of a version of the specification and drawings which I received from Mr. Squibbs during the afore-mentioned time period. During the afore-mentioned time period, I reviewed the application in order to provide comments and correction to Mr. Squibbs. During the afore-mentioned time period, I received an email message from Mr. Squibbs requesting me to review a revised version of the claims of the application prior to filing of the application. Attached as Exhibit G is a copy of the email message I received from Mr. Squibbs requesting the revised review. Exhibits D, F, and G were transmitted and/or received during the period between just prior to February 29, 2000 until October 17, 2000.

8. Attached as Exhibit H is a table comparing the elements of claims 19 and 36 with the corresponding conception identified in Exhibits A and B.

9. Exhibits A-G, which relate to the aforementioned conception of the claimed invention followed by diligence until a constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Actual dates of Exhibits A-B have been removed, but are prior to February 29, 2000. Exhibits D, F, and G, which relate to the aforementioned diligence until constructive reduction to practice are hereby incorporated by reference in their entirety herein, correspond to the invention broadly disclosed and claimed in the above-identified patent application. Further, company proprietary information has been removed from all

exhibits.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful false statements may jeopardize the validity of the application or any patent issuing thereon.

Date: 10/10/06

Rycharde Jeffery Hawkes

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, CO 80527-2400
Telephone: 703-684-1111
Facsimile: 970-898-0640

RAN/dll